# Computing on the Net: The History of Computing and the Vacuum.Net Theory

Nov 1, 2026 · J. Konstapel, Constable Research, Leiden · @hans konstapel

## Preface and method

This book sets the known theories of computing, from the abacus to inference-time scaling in 2026, next to a theory of computing derived from the Vacuum.Net Theory (Ninth Edition, 27 October 2026), and ends with a proposed device. Each theory is given with its inventor, its year, the devices that implemented it, and the place it takes in the net.

The Vacuum.Net theory of computing rests on four elements of the Foundations: the strand (N1), closure (N2), memory (N3) and scale with carry (N5, T7.1). Its empirical anchor is Sun et al. (2026), *What Effort Level Should I Choose? On the Economics of Reasoning Effort in LLM Agents*, arXiv:2610.34491.

Every statement carries one status, as in the Foundations (Part X):

| Status | Meaning in this book |
|---|---|
| Historical fact | Name, year, device; checked against a second source (record in the last section) |
| Proved | Follows from the axioms or from standard mathematics, with the proof given or cited |
| Computed | Recomputed with code for this book; the run is recorded |
| Corresponded | A declared link between an element of the net and an established theory; it can fail |
| Hypothesis | A testable claim this book puts forward; not yet tested |
| Open | A question the book cannot answer yet |

The language of the net is used throughout: a computation is a path that returns, closes or carries over. Where a source speaks of an agent that "decides" or "checks", the net reads a passage that closes or does not close.

## Part I — Before the machine

Before 1900 computing was a procedure carried out by hand or by gears: a fixed sequence of steps that returns a result. Three ideas from this period carry into the net: the procedure that ends (Euclid), the combination of a few signs into many (Llull, Leibniz), and the carry that passes a residue to the next position (the calculating machines).

| Year | Person | Theory or idea | Device |
|---|---|---|---|
| c. 300 BC | Euclid | Algorithm for the greatest common divisor (Elements VII): repeated subtraction until the remainder is zero | Hand, counting board |
| c. 1305 | Ramon Llull | Ars Magna: rotating discs combine a few concepts into all their pairings | Paper discs |

| Year | Person | Theory or idea | Device |
|---|---|---|---|
| 1623 | Wilhelm Schickard | Calculating clock: addition with mechanical carry | Gears |
| 1642 | Blaise Pascal | Pascaline: addition with carry across decimal wheels | Gears |
| 1672–1703 | G. W. Leibniz | Stepped reckoner; binary arithmetic (1703); calculus ratiocinator, reasoning as calculation | Stepped drum |
| 1822 / 1837 | Charles Babbage | Difference Engine; Analytical Engine with separate store and mill, conditional branching | Gears, punched cards (designed) |
| 1840 | Thomas Fowler | Balanced-ternary calculating machine | Wood |
| 1843 | Ada Lovelace | Notes on the Analytical Engine; first published program (Bernoulli numbers); symbols beyond number | Paper |
| 1847 / 1854 | George Boole | Algebra of logic with two values, 0 and 1 | Paper |

**Reading in the net.**

- *Euclid*: the algorithm stops when the remainder is zero. That is closure (N2) in its oldest form: the run ends where nothing is left over.
- *The carry*: Schickard and Pascal mechanised what T1 states as arithmetic: what a position does not keep passes on, whole, to the next. Every adding machine since is a device for N5.
- *Leibniz*: his principle of maximal richness from minimal means (Discours, 1686) is the design criterion the Foundations cite (III.1). He chose radix 2; radix economy places 3 nearest the optimum e (2.731 against 2.885 per unit of ln N; Foundations III.1).
- *Fowler*: the first machine to compute in the notation of the net, 118 years before the Setun.
- *Boole*: two values. The net works with three; a two-valued register has no value that differs from its reverse (Foundations T1 iii).

## Part II — The 1936 foundations

In 1936 four independent definitions of "computable" appeared and turned out to define the same class of functions. The Church–Turing thesis states that this class is everything a procedure can compute. Every later theory is measured against it.

| Year | Person | Theory | Core idea |
|---|---|---|---|
| 1928 | David Hilbert, Wilhelm Ackermann | Entscheidungsproblem | Is there a procedure that decides every statement of first-order logic? |

| Year | Person | Theory | Core idea |
|---|---|---|---|
| 1931 | Kurt Gödel | Incompleteness theorems | A consistent system rich enough for arithmetic contains true statements it cannot prove; statements refer to themselves through number codes |
| 1934 | Gödel, after Herbrand | General recursive functions | Functions defined by equations that refer back to themselves |
| 1936 | Alonzo Church | λ-calculus; undecidability | Computation as substitution: a function applied to itself; the Entscheidungsproblem has no solution |
| 1936 | Alan Turing | Turing machine | An unbounded tape, a head that reads one symbol, a finite table of states; the universal machine reads another machine's table |
| 1936 | Emil Post | Formulation 1 | A worker moving between boxes, marking and unmarking |
| 1936 | Stephen Kleene | μ-recursive functions; normal form | Every computable function = one search (μ) over a primitive recursive predicate |

No device implemented these in 1936; they are theories of what any device can do. The first stored-program machines ran in 1948–1949 (Part IV).

**Reading in the net.**

- *Gödel and Church*: both rest on a form referring to itself. The Foundations ground N1 in the same act: re-entry, the mark that contains itself (Part II of the Foundations; Spencer-Brown, Kauffman). Corresponded.
- *Turing's tape* is a strand with a head on it: one path, read one symbol at a time, written back. The difference: Turing's tape has no closure rule; a run continues until a halting state is reached from outside the tape's content. In the net, a run ends where its windows close (N2).
- *The halting problem* (Turing 1936) shows that no procedure decides for every machine whether it stops. In the net this reads: no procedure decides in advance for every path whether it closes. Sun et al. (2026) observe the same at small scale: a model cannot predict from the task description which effort level will close the task (self-routing, §3.4). Corresponded.
- *Kleene's μ* is a search that ends at the first success: the first passage that closes. Its structure equals the ladder policy of Sun et al.: try, and continue only if it did not close.

## Part III — Switching, information, neurons, feedback

Between 1937 and 1948 computing acquired its physical carrier (the switch), its measure (the bit) and its two biological models (the neuron and the feedback loop).

| Year | Person | Theory | Device |
|---|---|---|---|
| 1937 | Claude Shannon | Boolean algebra describes relay and switching circuits | Telephone relays |
| 1941 | Konrad Zuse | Z3: program-controlled binary floating-point computer | 2,600 relays, Berlin |

| Year | Person | Theory | Device |
|---|---|---|---|
| 1943–1944 | Tommy Flowers | Colossus: electronic code-breaking | Vacuum tubes, Bletchley Park |
| 1943 | Warren McCulloch, Walter Pitts | Logical calculus of neural activity: a neuron fires when its weighted inputs reach a threshold | Paper; later all neural networks |
| 1945 | J. Presper Eckert, John Mauchly | ENIAC: general-purpose electronic computer | About 17,500 vacuum tubes |
| 1948 | Claude Shannon | A Mathematical Theory of Communication: information as reduction of uncertainty, $H = -\Sigma\ p \log p$ | Telephone and radio channels |
| 1948 | Norbert Wiener | Cybernetics: control and communication through feedback | Anti-aircraft predictors, servomechanisms |

**Reading in the net.**

- *The switch* has two states. A relay or transistor stores no third value without extra circuitry; the binary era follows from the carrier, not from the mathematics. Balanced ternary requires a carrier with three stable states (Part X).
- *Shannon's entropy* counts the cost of distinguishing. In the net a distinction is a return of the strand (N1); a window of three trits carries $\log_2 27 = 4.75$ bits, and a closing window is one of 9, so a closure reports $\log_2 3 = 1.58$ bits less than an arbitrary window. Computed.
- *The threshold neuron* closes when its sum crosses a level. The net closes when its sum is zero modulo 3 (T2). Both are sum rules; the neuron's rule has one threshold and loses the sign, the net's rule keeps the residue and passes the carry on.
- *Wiener's feedback* is N3 in engineering form: each correction is taken relative to the first set point. Corresponded.

## Part IV — The stored program and the cellular automata

Von Neumann gave computing two architectures. The first, the stored-program computer, became every computer in use. The second, the cellular automaton, became the model for computation spread over space with only local rules, and is the closest established relative of the net.

| Year | Person | Theory | Device |
|---|---|---|---|
| 1945 | John von Neumann | First Draft of a Report on the EDVAC: program and data in one memory | EDVAC (design) |
| 1948 | F. C. Williams, Tom Kilburn | First stored program runs (21 June) | Manchester Baby, Williams tube |

| Year | Person | Theory | Device |
|---|---|---|---|
| 1949 | Maurice Wilkes | First practical stored-program service | EDSAC, mercury delay lines, Cambridge |
| 1940s–1966 | von Neumann, Stanisław Ulam | Cellular automaton; self-reproducing automaton with 29 states per cell | Paper; published by Burks, 1966 |
| 1970 | John Conway | Game of Life: two states, birth on 3 neighbours, survival on 2 or 3 | Paper and early computers |
| 1983–2002 | Stephen Wolfram | Elementary cellular automata, 256 rules; A New Kind of Science | Software |
| 2004 | Matthew Cook | Rule 110 is universal | Proof |

**Reading in the net.**

- *The von Neumann bottleneck*: program and data pass one channel between memory and processor. Every step re-reads from memory. Sun et al. measure the same at agent scale: each call re-reads the whole history, and input cost grows as n^1.82–1.92 in the number of calls (Part XIII). Computed.
- *The cellular automaton* is the established form of Foundations T10. The rewrite $R(t)_i = -(t_{i-1} + t_{i+1}) \bmod 3$ is a one-dimensional cellular automaton with three states and a radius-1 neighbourhood. It is additive (linear over $\mathbb{Z}_3$). Additive automata are fully described by polynomial algebra (Martin, Odlyzko and Wolfram 1984) and are not universal on their own. Proved.
- *Life and Rule 110* are universal and non-linear. The net becomes non-linear through the carry between scales (T1, T7.1): c(1) + c(1) = 0 but c(1 + 1) = 1. Computed. Whether R plus carry plus memory is universal is open (Part XVI).
- *Self-reproduction*: von Neumann's automaton copies its own description. The eigenform O(A) = A of the Foundations (II.2) is the fixed-point form of the same idea. Corresponded.

## Part V — Languages, complexity, algorithmic information

After 1950 the question moved from *what* can be computed to *with what memory* (automata and languages), *at what cost* (complexity) and *how short* a description can be (algorithmic information).

| Year | Person | Theory | Core idea |
|---|---|---|---|
| 1951 / 1956 | Stephen Kleene | Regular events (regular expressions) | What a machine with finite memory recognises |
| 1956 / 1959 | Noam Chomsky | Hierarchy of grammars | Four levels: regular, context-free, context-sensitive, unrestricted; each level adds memory (none, stack, bounded tape, unbounded tape) |

| Year | Person | Theory | Core idea |
|---|---|---|---|
| 1959 | Michael Rabin, Dana Scott | Finite automata and their decision problems | Non-deterministic automata; equal power to deterministic ones |
| 1965 | Juris Hartmanis, Richard Stearns | Computational complexity | Time and space as resources of a Turing machine |
| 1971 / 1972 / 1973 | Stephen Cook; Richard Karp; Leonid Levin | NP-completeness | Problems whose solution is easy to check but, as far as known, hard to find |
| 1960–1969 | Ray Solomonoff, Andrey Kolmogorov, Gregory Chaitin | Algorithmic information | The complexity of a string = the length of its shortest program |
| 1975 | Gregory Chaitin | Ω, the halting probability | A number that is well defined but not computable |

Devices: these theories are implemented in compilers (lexers are finite automata, parsers are stack machines), in solvers for NP-hard problems, and in compression software.

**Reading in the net.**

- *Chomsky's levels are levels of memory.* The net places memory in the shape of the path (N3). A path whose windows all close needs only its last window: 9 possible states, a finite automaton (T8). A path with carry needs a stack per scale. Proved for the reduction in Part XIII: each scale holds at most two open passages.
- *NP: easy to check, hard to find.* Sun et al. show this asymmetry at agent scale. Checking closure is cheap (a test runs once); finding it is expensive. The ladder policy spends on finding only where the cheap search did not close. Corresponded.
- *Chaitin* is the comparison metric the Foundations adopt (III.1, IX.6): a theory is a program shorter than its data. Vacuum.Net claims five axioms for its range; Part XIV counts the axioms of the other theories in the same way.
- *Ω* is the probability that a random program halts. Its net reading: the probability that a random path closes. For one window under uniform trits this is exactly 1/3 (Foundations T6). Proved. For whole paths it is open.

## Part VI — Meaning and types

From 1969 a second line of theory asked what a program *means*. The answer that took hold: the meaning of a recursive program is the fixed point of the operation that defines it. A parallel line found that programs and proofs are the same objects.

| Year | Person | Theory | Core idea |
|---|---|---|---|
| 1969–1971 | Dana Scott, Christopher Strachey | Domain theory; denotational semantics | A recursive definition means the least fixed point of a continuous function on a domain |
| 1934 / 1958 / 1969 | Haskell Curry; William Howard | Curry–Howard correspondence | A type is a proposition; a program of that type is its proof |
| 1972–1975 | Per Martin-Löf | Intuitionistic type theory | Types depending on values; the basis of proof assistants (published version 1975) |
| 2006–2013 | Vladimir Voevodsky; Univalent Foundations Program | Homotopy type theory | Types as spaces, equalities as paths; univalence: equivalent structures are equal |

Devices: proof assistants (Coq/Rocq, Agda, Lean) and typed functional languages (ML, Haskell) implement these theories on ordinary binary hardware.

**Reading in the net.**

- *The least fixed point* is the eigenform of the Foundations (II.2): O(A) = A. On the register, the fixed points of the rewrite R are exactly the rings that close in every window (T10). The Scott semantics of a program and the closure of a ring are the same condition. Corresponded; on the register proved.
- *Curry–Howard*: a proof is a computation that ends. In the net a proof is a path that closes. A type that has no closed path is a proposition with no proof.
- *Homotopy type theory* reads equality as a path between points. The net reads every distinction as a path that returns (N1). Both make the path primary and the point secondary. The earlier lead on modal homotopy type theory (Schreiber) fits here. Corresponded.
- Difference: these theories assume an unlimited supply of steps. The net counts the cost of every step against what it leaves open (Part XIII).

## Part VII — Concurrency and interaction

When several processes run at once, a computation is no longer one path from input to output but a pattern of exchanges. Four theories describe this; all four take the trace — the ordered record of events — as the object.

| Year | Person | Theory | Core idea |
|---|---|---|---|
| 1962 | Carl Adam Petri | Petri nets | Places hold tokens; a transition fires when its input places are filled |
| 1973 | Carl Hewitt, Peter Bishop, Richard Steiger | Actor model | Everything is an actor that receives messages, creates actors, sends messages |

| Year | Person | Theory | Core idea |
|---|---|---|---|
| 1978 / 1985 | Tony Hoare | Communicating Sequential Processes | A process is the set of its traces; processes synchronise on shared events |
| 1980 / 1989–1992 | Robin Milner (with Joachim Parrow, David Walker) | CCS; π-calculus | Processes exchange names over channels; channels themselves can be passed on |

Devices: Petri nets in workflow and manufacturing control; actor-style processes in Erlang (telecom switches, designed independently of Hewitt) and Akka; CSP in the Transputer and its language occam (1980s) and in Go's channels; π-calculus in protocol verification.

**Reading in the net.**

- *The trace is the strand.* Hoare's process is the set of its traces; the net's computation is its path (N1). Corresponded.
- *Synchronisation is a meeting.* Two processes that exchange an event and continue form the pair $(x, -x)$ of T7: a meeting that closes with carry 0. A handshake leaves nothing over.
- *The Petri transition is a closure rule.* It fires when all inputs are present; the net's window closes when its sum is zero. Both are local conditions that select which configuration persists.
- *Milner's mobility* (passing a channel on) is N5: a closed exchange becomes one passage on the next scale and can itself be passed. Corresponded.
- Difference: none of these theories counts the cost of carrying the trace. In practice the trace grows without limit. The net closes it and keeps only the carry (Part XIII).

## Part VIII — The physics of computation

From 1961 computing was treated as a physical process with a cost in energy. Three lines followed: reversible computing (what is lost when information is erased), quantum computing (what a physical system computes by evolving), and analog computing (computation over continuous quantities).

| Year | Person | Theory | Device |
|---|---|---|---|
| 1931 | Vannevar Bush (with Harold Hazen) | Differential analyzer | Mechanical integrators, MIT |
| 1941 | Claude Shannon | General Purpose Analog Computer (GPAC) | Model of the differential analyzer |
| 1961 | Rolf Landauer | Erasing one bit costs at least $kT \ln 2$ of heat | Principle; approached experimentally by Bérut et al. (2012) |

| Year | Person | Theory | Device |
|---|---|---|---|
| 1973 | Charles Bennett | Any computation can be made logically reversible | Model |
| 1980 / 1982 | Tommaso Toffoli; Edward Fredkin and Toffoli | Reversible gates; conservative logic | Gate designs |
| 1980 / 1982 / 1985 | Paul Benioff; Richard Feynman; David Deutsch | Quantum Turing machine; simulating physics with physics; universal quantum computer | Theory |
| 1989 | Lenore Blum, Michael Shub, Stephen Smale | Computation over the real numbers | Model |
| 1994 / 1996 | Peter Shor; Lov Grover | Quantum factoring; quantum search | Algorithms |
| 2019 | Google (Arute et al.) | Sampling beyond classical reach | Sycamore, 53 working qubits |

**Reading in the net.**

- *Landauer and loss.* Physics T6 states that loss has an address: every irreversible discharge occurs at a mesh. Landauer places the cost of computation exactly there: at the erasure. In the net an erased passage is a window that closes; the closure with carry 0 (a standing, a meeting) is the point where information leaves the path. Corresponded. Consequence for Part XIII: the cheap memory of the net is bought with erasure, and erasure costs heat.
- *Reversibility.* Bennett's reversible computer keeps its history to avoid the heat. That is the full-history agent of Sun et al.: no loss, cost $n^2$ in reading. The net sits between: it keeps the carry and erases the closed. Corresponded.
- *Quantum computing* rests on two-valued systems with superposition (qubits) and on unitary, reversible evolution. Physics T1 places Dirac's equation in Rowlands' nilpotent form, $Q^2 = 0$ on the mass shell. A three-level system (a qutrit) is the quantum carrier of the trit; qutrit processors exist in laboratories. The net's reading of measurement as closure is a correspondence (C3), not a quantum algorithm.
- *Analog computing* drops the discrete step. The net keeps it: a trit is a residue, not a magnitude.

## Part IX — Closure and self-reference

A separate tradition, largely outside computer science, made closure the central concept: a system that produces what produces it. This is the line from which the Vacuum.Net Theory itself descends.

| Year | Person | Theory | Core idea |
|---|---|---|---|
| 1958 / 1991 | Robert Rosen | (M,R)-systems; Life Itself | An organism is closed under efficient causation; such systems cannot be fully simulated by a Turing machine (Rosen's claim) |

| Year | Person | Theory | Core idea |
|---|---|---|---|
| 1962 | Herbert Simon | The Architecture of Complexity | Stable intermediate forms; nearly decomposable hierarchies (the watchmaker parable) |
| 1969 | George Spencer-Brown | Laws of Form | Draw a distinction; calling twice is calling once, crossing twice is not crossing; re-entry |
| 1972/1973 | Humberto Maturana, Francisco Varela | Autopoiesis | A living system is a network of processes that produces its own components |
| 1976 | Heinz von Foerster | Eigenforms ("Objects: tokens for (eigen-)behaviors" | A stable object is the fixed point of a recursive operation |
| 2002 / 2007 | Peter Rowlands, Bernard Diaz | Universal alphabet and rewrite system; nilpotent Dirac operator | A rewrite system that starts from zero and generates the algebra of physics |
| 2005 | Louis Kauffman | EigenForm | O(A) = A; i = [+1, −1]η as the oscillation of re-entry |

Devices: none dedicated. These theories run as simulations on binary computers, and in Rosen's view they describe what such computers cannot be.

**Reading in the net.**

- *Spencer-Brown and Kauffman* are the declared ground of N1 and N2 (Foundations Part II). The net adds the arithmetic: the mark becomes a trit, re-entry becomes a return of the strand, and closure becomes a sum that is zero modulo 3.
- *Simon's watchmaker* is N5 in its organisational form. Tempus, whose unfinished watch falls apart at each interruption, loses everything; Hora, who builds stable sub-assemblies, loses only the part in hand. The net's reduction keeps the closed sub-assemblies as carry and loses only the open window (Part XIII). Corresponded.
- *Rosen's closure* is the form of the Foundations' open question IX.2: which rewrite dynamics is its own fixed point. On the register this is solved (T9, T10); for the whole net it is open.
- *Rowlands and Diaz* are the nearest computing relative: a rewrite system from zero to physics. The net shares the nilpotent condition (Physics T1, C3) and adds the ternary register and the carry.

## Part X — Ternary computing

Balanced ternary has been built in hardware twice at scale and is returning in two places: low-power logic and the weights of large language models. It is the notation of the net (Foundations III.2).

| Year | Person or group | Work | Device |
|---|---|---|---|
| 1840 | Thomas Fowler | Balanced-ternary calculating machine | Wood |
| 1958 | Nikolay Brusentsov, Sergei Sobolev (Moscow State University) | Setun: balanced-ternary computer, 18-trit words; about 50 built at Kazan, 1959–1965 | Pairs of ferrite cores with diodes |
| 1970 | Brusentsov | Setun-70 | Ferrite cores |
| 1969 / 1997 | Donald Knuth | The Art of Computer Programming, Vol. 2, §4.1: balanced ternary as "perhaps the prettiest number system of all" | Text |
| 2019 | Kim Kyung Rok (UNIST), funded by Samsung | Ternary CMOS on a full silicon wafer (Nature Electronics) | Silicon; third state from a constant band-to-band tunnelling current |
| 2024 | Ma et al. (Microsoft Research) | BitNet b1.58: large language models with weights in {−1, 0, 1} | Software on binary GPUs |
| 2023 / 2025 | Huawei | Ternary logic gate circuit, patent application CN119652311A (filed 2023, published March 2025) | Silicon (application) |

**Reading in the net.**

- *The Setun and the net use the same digits.* Foundations III.2 lists four properties that only balanced ternary has among radix-3 notations: no minus sign, carry-free single-trit multiplication, truncation equals rounding, shift equals scale step. Proved there.
- *Why binary won.* The switch has two stable states (Part III). The Setun obtained three from pairs of cores. The 2019 UNIST work obtains the third state from a constant off-state tunnelling current. The obstacle was the carrier, not the arithmetic.
- *BitNet b1.58* shows that the {−1, 0, 1} alphabet carries the knowledge of a language model. The name counts $\log_2 3 = 1.58$ bits per weight. Computed. In BitNet the trits are weights, not closures: the model is ternary, the computation around it is not.
- *What no ternary device has had yet* is the closure rule as a hardware operation: a register that detects a closed window and replaces it by its carry. That is the device proposed in Part XV.

## Part XI — Learning machines

Learning machines compute by adjusting weights instead of following a written program. Since 2024 a second dial has appeared beside the size of the model: how much computation is spent at the moment of answering. Sun et al. (2026) measure what that dial actually buys.

| Year | Person | Theory or system | Device |
|---|---|---|---|
| 1958 | Frank Rosenblatt | Perceptron | Mark I Perceptron, Cornell Aeronautical Laboratory |
| 1969 | Marvin Minsky, Seymour Papert | Perceptrons: limits of one layer | Analysis |
| 1986 | David Rumelhart, Geoffrey Hinton, Ronald Williams | Backpropagation | Workstations |
| 2017 | Vaswani et al. | Transformer: attention over the whole context | GPUs, TPUs |
| 2024 | Snell et al.; OpenAI o1 (September 2024) | Test-time compute: reasoning before answering | GPU clusters |
| 2026 | Sun, Kwon, Hasan, Zhu, Zhou, Bianchi | The economics of reasoning effort in LLM agents | 8 models, 4 benchmarks |

**What Sun et al. measured.**

| Finding | Number |
|---|---|
| Highest effort on tasks every level solves | up to 42× the cost of the lowest |
| GPT-5.6-sol, SWE-Bench Pro, low → xhigh | $0.27 → $1.24 per task; 14.7 → 28.4 calls |
| GPT-5.6-luna, reasoning off ("none") against low | 21.5 against 5.4 calls; 4.58× the cost |
| Reference fix read from git history, GPT-5.6-sol | 15.9% (low) → 75.0% (xhigh) |
| Corrected environment, highest effort | lower solve rate for 8 of 8 models; sign test p = 0.008 |
| Ladder (lowest effort, escalate after verified failure), median saving | 25.3% AIME, 41.5% GPQA, 15.5% corrected SWE-Bench Pro |
| User waiting time | 58–98% of the full cost |

**Reading in the net.** The correlation of these findings with N2, N3, N5 and T7.1 is worked out in Part XIII. In short: the transformer re-reads its whole context at every step, which is the full-history computer of Bennett (Part VIII) at agent scale; its cost grows quadratically. Reasoning effort does not close earlier on easy tasks; the path continues past closure because the closure test lies outside the path (the evaluator). The ladder policy is the scale rule of the net.

## Part XII — Unconventional substrates

Since 1989 computation has been demonstrated in molecules, membranes, spiking circuits, resistive memory and rewriting graphs. Each substrate brings its own native operation; the question for the net is which substrate has a native closure.

| Year | Person or group | Theory | Device |
|---|---|---|---|
| 1989 | Carver Mead | Neuromorphic engineering (Analog VLSI and Neural Systems) | Analog silicon |

| Year | Person or group | Theory | Device |
|---|---|---|---|
| 1994 | Leonard Adleman | DNA computing: a 7-vertex Hamiltonian path solved in a test tube | DNA strands |
| 1998 | Gheorghe Păun | Membrane computing (P systems) | Model of nested compartments |
| 1971 / 2008 | Leon Chua; Strukov, Snider, Stewart, Williams (HP Labs) | Memristor: resistance that remembers the charge that passed | Titanium-dioxide films |
| 2014 / 2017 | IBM (Merolla et al.); Intel | TrueNorth; Loihi spiking processors | Digital neuromorphic chips |
| 2020 | Stephen Wolfram, Jonathan Gorard, Max Piskunov | Wolfram Physics Project: hypergraph rewriting | Software |

**Reading in the net.**

- *The memristor* is the closest physical analogue of N3: its present state is a function of the charge that has passed, the history carried as shape. Several groups have built ternary logic from memristors (for example patent CN111046617A). Corresponded.
- *DNA computing* computes by hybridisation: a strand finds its complement and closes. Complementary pairing is the meeting (x, −x) of T7, closing with carry 0. Corresponded.
- *Membranes* nest compartments inside compartments, each with local rules: N5 as architecture.
- *Spiking net*works with spikes in {−1, 0, +1} have been trained (Ternary Spike, AAAI 2024): the trit alphabet in neuromorphic form.
- *Wolfram's hypergraphs* select rules by whether known physics emerges, an external criterion (Foundations IX.6). The net selects by closure, a criterion inside the rule (N2).

## Part XII-A — The complete catalogue

The four catalogues below hold 742 distinct theories, methods and devices of computation, from the Lebombo bone (c. 43,000 years ago) to the reasoning-effort study of October 2026. Each entry was found by a respondent and passed by an independent judge; none was admitted on the respondent's word alone.

**Method (respondent–judge protocol).** Four respondents each searched one domain and listed every item they could find, with person, year, core idea and device. Eleven judges, who had not seen the respondents' reasoning, checked every item against an opened source and returned ACCEPT, CORRECT (with the corrected field) or REJECT. Only judged items enter the catalogue; every correction is applied; every row carries the judge's source.

| Catalogue | Period or domain | Candidates | Accepted | Of which corrected | Rejected | Already in an earlier catalogue |
|---|---|---|---|---|---|---|
| 1 | Prehistory to 1900 | 112 | 112 | 29 | 0 | 0 |

| Catalogue | Period or domain | Candidates | Accepted | Of which corrected | Rejected | Already in an earlier catalogue |
|---|---|---|---|---|---|---|
| 2 | 1900–1965 | 232 | 232 | 16 | 0 | 0 |
| 3 | Theoretical computer science, 1965–2026 | 224 | 219 | 28 | 0 | 5 |
| 4 | AI, learning machines, physical and unconventional computing | 228 | 179 | 22 | 0 | 49 |
| **Total** | | **796** | **742** | **95** | **0** | **54** |

Computed: 112 + 232 + 219 + 179 = 742; 796 − 54 = 742.

**Re-check.** For some entries the judge could confirm a field only from search-result listings, because the shared limit on opened pages was reached. A second judge re-checked all 113 such entries against opened pages on 1 November 2026: 111 confirmed, 2 corrected (CLARION 1997, BrainScaleS 2010), none left open. No entry in the catalogues rests on search listings alone.

Parts I–XII above treat about 55 of these theories at length; the catalogues give the rest in table form, each theme followed by its reading in the net.

## Catalogue 1 — Prehistory to 1900

112 items accepted from 112 candidates, 29 corrected, 0 rejected, judged under the respondent–judge protocol.

## Prehistory and number systems

| Date | Method, system or device | Person(s) or culture | Core idea | Medium | Source |
|---|---|---|---|---|---|
| c. 43,000–20,000 BP (Ishango dating disputed, 11–44 ka) | Tally marks: Lebombo and Ishango bones | anonymous | One-to-one tallying; arithmetic reading of the Ishango groupings is contested | notched bone | link |
| c. 8000–3000 BCE | Clay tokens and bullae | Neolithic Near Eastern farmers (Zagros); bullae c. 3500 BCE (Uruk, Susa) | Concrete counters for goods; precursor of numerals and writing; replaced by tablets c. 3100–3000 BCE | clay tokens, envelopes | link |
| c. 2000–1600 BCE | Sexagesimal place value and reciprocal tables | Old Babylonian scribes | Base-60 positional numbers; division as multiplication by tabulated reciprocals | clay tablets | link |
| c. 1800–1600 BCE | Square root of 2 (YBC 7289) | Old Babylonian scribe | 1;24,51,10 = 1.41421296; method unknown (copied from a constants table or iterative averaging, hypothesis) | clay tablet | link |
| c. 1550 BCE (copy of a text of c. 1850–1800 BCE) | Doubling multiplication and unit fractions (Rhind papyrus) | Ahmes | Multiply by doubling and adding; 2/n table for odd n 3–101 | papyrus | link |
| c. 475 BCE onward | Counting rods (rod numerals) | Chinese officials and mathematicians | Decimal place value, blank for zero; red positive, black negative | bamboo or bone rods on a board | link |
| c. 300 BCE–400 CE | Counting board, Salamis tablet, Roman hand abacus | Greek and Roman reckoners | Pebbles (calculi) on ruled lines give place-value arithmetic | marble board, bronze hand abacus | link |

| Date | Method, system or device | Person(s) or culture | Core idea | Medium | Source |
|---|---|---|---|---|---|
| c. 1st c. BCE (concept); zero attested 3rd c. CE; to 900 CE | Vigesimal positional notation with zero | Maya astronomer-priests; earliest Long Count date 36 BCE (Chiapa de Corzo, possibly pre-Maya) | Base-20 place value with a zero sign for calendar computation | stone stelae, codices | link |
| c. 600–1532 CE | Quipu (khipu) and yupana | Wari (earliest), then Inca quipucamayocs | Knotted cords record decimal values; yupana arithmetic and base (10 or 40) disputed, attested via Guaman Poma (c. 1615) | knotted cords, tray board | link |
| undated; c. 900–1500 CE unverified (disputed) | Nepohualtzintzin | Mesoamerican reckoners; known via Esparza Hidalgo's 20th-c. reconstruction | Vigesimal bead abacus, 13 rows x 7 beads = 91 | reconstructed bead frame | link |
| c. 1200–1250 CE (copy of material 300–400 years older) | Dresden Codex Venus and eclipse tables | Maya scribes | Tabulated long-count cycles predict Venus and eclipses | amate (fig-bark) paper codex | link |
| traditional, oral; Oyo late 1400s (Johnson 1921) | Ifá | Yoruba babalawo | Divination system with a binary combinatorial code: 256 odu = 16 x 16 four-line signs; not a calculation method | palm nuts (ikin), opele chain, opon tray | link |

In the net: a row of tally notches is the strand, one ordered path in which each mark follows the previous one (N1). In the place-value systems of base 60, 20 and 10, what a position does not keep passes as carry to the next scale (N5); a full position rolls over as a ring (x,x,x, carry ±1), and the red and black rods hold a meeting (x,−x, carry 0) in which a positive and an equal negative cancel. Corresponded.

## Ancient algorithms and formal systems (Greece, India, China)

| Date | Method, system or device | Person(s) or culture | Core idea | Medium | Source |
|---|---|---|---|---|---|
| c. 6th–4th c. BCE (often c. 350 BCE) | Aṣṭādhyāyī as formal grammar | Pāṇini | 3,983 ordered rewrite rules with metarules (paribhāṣā) generate Sanskrit | Sanskrit sūtras | link |
| c. 350 BCE | Syllogistic | Aristotle | Valid inference forms between categorical propositions; 14 moods in three figures (Prior Analytics) | text | link |
| c. 300 BCE | Euclidean algorithm | Euclid | Repeated subtraction gives the greatest common divisor (Elements VII.1–2, X.2–3) | text (Elements) | link |
| c. 3rd–2nd c. BCE | Chandaḥśāstra (binary prosody) | Pingala; meru-prastāra explicit in Halāyudha (10th c. CE) | Metres enumerated as binary laghu/guru strings | Sanskrit sūtras | link |
| c. 250 BCE | Stoic propositional logic | Chrysippus | Five indemonstrable schemata over if, not-both and or | text | link |
| c. 250 BCE | Sand Reckoner | Archimedes | Orders and periods with base 10^8 reach 10^(8x10^16): finite, very large | text | link |
| c. 240 BCE | Sieve of Eratosthenes | Eratosthenes (reported by Nicomachus, 2nd c. CE) | Strike out multiples in turn; the primes remain | text, procedure | link |
| c. 205–87 BCE (disputed); wreck c. 70–60 BCE | Antikythera mechanism | anonymous Greek (Hipparchan tradition) | Gear trains model lunar, solar and eclipse cycles | bronze geared device | link |
| c. 200 BCE–100 CE | Nine Chapters (Jiuzhang suanshu), fangcheng | anonymous; commentary Liu Hui (263) | Elimination on the counting board solves linear systems | rods on board | link |

| Date | Method, system or device | Person(s) or culture | Core idea | Medium | Source |
|---|---|---|---|---|---|
| c. 60 CE | Square-root method and programmable cart | Heron of Alexandria | Iterative square-root averaging (Metrica); cord on axle drums with pegs sequences cart motions | text; rope-drum automaton | link |
| 263 (traditional); c. 460s | π by polygon doubling | Liu Hui; Zu Chongzhi | Liu: 3.14159 with a 3,072-gon; Zu: 3.1415926 < π < 3.1415927 | rods, text | link |
| c. 3rd–5th c. CE | Remainder problem (Chinese remainder theorem) | Sunzi | Recover a number from its remainders modulo coprime moduli | text (Sunzi Suanjing) | link |
| 499 | Kuṭṭaka ("pulveriser") | Āryabhaṭa; named and detailed by Bhāskara I (c. 629) | Recursive division solves linear Diophantine equations | Sanskrit verse | link |
| 628 | Arithmetic of zero and negatives | Brahmagupta | Explicit rules for zero and negative numbers (incl. 0/0 = 0) | Sanskrit text | link |
| 628; c. 950–1000; 1150 | Bhāvanā and chakravāla | Brahmagupta; Jayadeva; Bhāskara II | Composition rule and cyclic algorithm solve Pell's equation | Sanskrit text | link |
| manuscript 799–1102 CE (C14, 2024); composition disputed | Bakhshali manuscript | anonymous Indian | Decimal place value with dot-zero; one-step correction formula for square roots | birch bark | link |

In the net: the Euclidean algorithm, the sieve and the kuṭṭaka run along one ordered path (N1) and stop when nothing is left over; that stop is closure (N2), as a window of three trits closes when its sum is 0 mod 3. The remainder problem reads a number as a set of residues, each a response fixed by its modulus as context (N4), and Brahmagupta's negatives make the meeting (x,−x, carry 0) explicit, with zero as the standing closure. Corresponded.

## Islamic world and medieval period

| Date | Method, system or device | Person(s) or culture | Core idea | Medium | Source |
|---|---|---|---|---|---|
| 2nd c. BCE (Hipparchus, disputed); treatises 4th c.–c. 815 CE | Astrolabe | Greek origin; Theon, Philoponus; al-Fazārī and Islamic makers | Stereographic projection solves positional astronomy and time problems | brass disc | link |
| c. 820–825 | Hindu reckoning and algebra | al-Khwārizmī | Decimal-numeral procedures ("algorism"); systematic al-jabr equation solving | Arabic text, dust board | link |
| c. 850 | Frequency-analysis cryptanalysis | al-Kindī | Letter frequencies break substitution ciphers | Arabic manuscript | link |
| c. 850 | Automatic flute player | Banū Mūsā brothers | Pinned rotating cylinder stores an exchangeable musical program | water-driven pinned barrel | link |
| 952 | Pen-and-paper arithmetic with decimal fractions | al-Uqlīdisī | Written Hindu-Arabic arithmetic replaces the dust board; early decimal fractions | paper | link |
| c. 980 | Abacus with apices | Gerbert of Aurillac (Sylvester II) | Counters marked with nine Hindu-Arabic numerals on a column board, no zero | counting board | link |
| c. 1060 | Hexagram ordering | Shao Yong | 64 hexagrams in a sequence equivalent to binary counting | diagram (I Ching) | link |
| c. 1080; 1326 | Equatorium and Albion | Ibn al-Zarqālluh; Richard of Wallingford | Scale model computes planetary positions without tables | brass plates | link |
| 1088 model; 1090 bronze; 1094 tower | Astronomical clock tower | Su Song | Water-wheel escapement drives armillary sphere and celestial globe | hydromechanical tower | link |

| Date | Method, system or device | Person(s) or culture | Core idea | Medium | Source |
|---|---|---|---|---|---|
| c. 1085–1145 (first depiction); 1592 | Suanpan (bead abacus) | Chinese reckoners; Cheng Dawei (Suanfa tongzong) | Bead abacus with rhymed algorithms (root extraction not verified) | bead frame | link |
| 1202 | Liber Abaci | Leonardo of Pisa (Fibonacci) | Hindu-Arabic algorithms for European commerce | Latin text | link |
| 1206 | Castle clock and programmable automata | al-Jazarī | Cams and pegs set sequences for clock and musical automata | water clock, camshaft | link |
| 1247 | Dayan method and polynomial root extraction | Qin Jiushao | General remainder algorithm; Horner-type numerical root solving | rods (Shushu Jiuzhang) | link |
| c. 1274–1308 | Ars (Ars Magna) | Ramon Llull | Rotating concentric figures combine concepts into propositions | paper volvelles | link |
| 1303 | Four-unknowns method | Zhu Shijie | Elimination of polynomial systems to one unknown | rods (Siyuan yujian) | link |
| c. 1350–1425 | Kerala series | Mādhava of Sangamagrāma | Infinite series for π, sine and arctangent with correction terms | Sanskrit and Malayalam texts | link |
| 1416; 1424 | Plate of conjunctions; plate of heavens; π | Jamshīd al-Kāshī | Interpolation device for conjunction times; equatorium for positions; 2π to 16 decimals via a 3x2^28-gon | instruments, text | link |

In the net: the pinned cylinders of the Banū Mūsā and al-Jazarī hold one ordered path (N1) in which the first passage of the pins fixes the sequence and each later turn reads relative to it (N3). Shao Yong's ordering lists the 64 hexagrams as binary words, and the move from dust board to written numerals keeps the carry between positions on paper (N5). Qin Jiushao's remainder algorithm and Zhu Shijie's elimination stop when no unknown is left over, which is closure (N2). Corresponded.

## Renaissance and early modern period (to 1800)

| Date | Method, system or device | Person(s) or culture | Core idea | Medium | Source |
|---|---|---|---|---|---|
| 1467 | Cipher disk | Leon Battista Alberti | Two concentric disks give polyalphabetic substitution | disk | link |
| c. 1580 | Prosthaphaeresis | Wittich; Bürgi; Tycho Brahe | Trigonometric identities turn multiplication into addition | tables | link |
| 1585 | Decimal fractions (De Thiende) | Simon Stevin | Systematic decimal fractions for practical computation | printed booklet | link |
| 1597 | Geometric and military compass (sector) | Galileo; Thomas Hood | Proportional scales on hinged arms solve proportion problems | brass sector | link |
| c. 1601–1605 (undated) | Binary arithmetic | Thomas Harriot | Arithmetic in base 2; unpublished, known from the 1920s | manuscript | link |
| 1605/1623 | Biliteral cipher | Francis Bacon | Letters encoded as 5-symbol two-valued strings | text | link |
| 1614; 1620 | Logarithms | John Napier; Jost Bürgi | Multiplication reduced to addition through tables | printed tables | link |
| 1617 | Napier's bones (Rabdologia) | John Napier | Lattice multiplication on numbered rods | wood, ivory or metal rods | link |
| 1617/1624 | Common (base-10) logarithms | Henry Briggs | Decimal logarithm tables to 14 places | printed tables | link |
| 1620; c. 1622 | Gunter's scale and slide rule | Edmund Gunter; William Oughtred | Logarithmic scales add lengths to multiply | wood or brass rule | link |
| 1623 | Calculating clock | Wilhelm Schickard | Napier rods plus geared 6-digit adder with single-tooth carry | gear mechanism | link |
| 1642 | Pascaline | Blaise Pascal | Gear adder with gravity-driven carry; subtraction by nines' complement | gear mechanism | link |
| 1650 | Arca Musarithmica | Athanasius Kircher | Combinatorial tablets compose four-part music | wooden chest of tablets | link |

| Date | Method, system or device | Person(s) or culture | Core idea | Medium | Source |
|---|---|---|---|---|---|
| 1651/1655 | Reasoning as computation | Thomas Hobbes | "Reckoning (that is, adding and subtracting)" of names (Leviathan, De Corpore) | text | link |
| c. 1666–1716 | Characteristica universalis / calculus ratiocinator | G. W. Leibniz | Universal symbolic language; disputes settled by calculating | text, project | link |
| 1666 | Adding machine | Samuel Morland | Dial-wheel adder for sterling currency, no automatic carry | brass dials | link |
| 1669/1690 | Newton–Raphson iteration | Isaac Newton; Joseph Raphson | Iterative correction of polynomial roots without derivatives; tangent form for general equations is Simpson (1740) | text | link |
| 1670 | Study of number bases (Mathesis biceps) | Juan Caramuel y Lobkowitz | Treatment of bases 2–10, 12 and 60 | text | link |
| 1672–1694 | Stepped reckoner | G. W. Leibniz | Stepped drum: multiplication by repeated addition with shifting | gear mechanism | link |
| 1676/1711 | Finite-difference interpolation | Isaac Newton | Polynomial interpolation from tabulated differences | text | link |
| 1679/1703 | Binary arithmetic | G. W. Leibniz (Bouvet, I Ching) | Base-2 arithmetic, "Explication de l'Arithmétique Binaire" | text | link |
| 1683 | Elimination, resultants and determinants | Seki Takakazu | Determinants eliminate unknowns (wasan) | written symbolic notation (bōsho), replacing sangi rods | link |
| 1709 | Pinwheel calculator | Giovanni Poleni | Variable-tooth pinwheel for multiplication | wooden gear mechanism | link |

| Date | Method, system or device | Person(s) or culture | Core idea | Medium | Source |
|---|---|---|---|---|---|
| 1725; 1728 | Punched-paper and card loom control | Basile Bouchon; Jean-Baptiste Falcon | Perforated paper or cards select warp threads | punched paper, cards | link |
| c. 1740–1748 | Automated loom | Jacques de Vaucanson | Perforated drum guides hooks to warp yarns: partial automation | punched cylinder | link |
| 1768 | Euler method | Leonhard Euler | Stepwise tangent approximation of ODE solutions | text | link |
| 1768–1774 | Writer automaton | Pierre and Henri-Louis Jaquet-Droz; Jean-Frédéric Leschot | Wheel of reorderable letter pieces selects cams: programmed text | clockwork automaton | link |
| 1774 | Calculating machine | Philipp Matthäus Hahn | Cylindrical stepped-drum calculator, small series | gear mechanism | link |
| 1775 | Stanhope Demonstrator | Charles Stanhope | Slides solve numerically definite syllogisms and probability problems | wood and brass device | link |
| 1779; 1795 | Lagrange interpolation | Edward Waring (first); Lagrange (republished) | Explicit interpolating polynomial through given points | text | link |
| 1784/1786 (disputed) | Difference-engine concept | Johann Helfrich Müller | Proposed machine to tabulate polynomials by differences | design text | link |
| 1791–1801 | Tables by division of mental labour | Gaspard de Prony | Three-tier hierarchy of about 90 human computers using differences | people, paper | link |

In the net: in the adders of Schickard and Pascal what a wheel does not keep passes as carry to the next wheel (N5); a full wheel rolling from 9 to 0 is the ring (x,x,x, carry ±1), and Pascal's subtraction by nines' complement is the meeting (x,−x, carry 0). Morland's dials leave the carry open, so the tens stay with the operator. Logarithms and the slide rule move multiplication onto one ordered scale (N1), and the punched paper of Bouchon and Falcon fixes on first passage the order that every later pass reads relative to it (N3). Corresponded.

## Nineteenth century

| Date | Method, system or device | Person(s) or culture | Core idea | Medium | Source |
|---|---|---|---|---|---|
| 1804 | Jacquard loom | Joseph Marie Jacquard | Chain of punched cards controls patterns: stored program for weaving | punched cards | link |
| 1805–1810 | Least squares, FFT-type algorithm, elimination notation | C. F. Gauss | Least squares publ. 1809 (Legendre 1805); FFT-type c. 1805, publ. 1866; bracket notation for common elimination (1810) | text | link |
| 1820 patent; production 1851 | Arithmometer | Charles Xavier Thomas de Colmar | First commercially successful stepped-drum calculator | gear mechanism | link |
| 1822–1842 | Difference Engine | Charles Babbage | Method of differences tabulates polynomials, with printing | gear mechanism (demonstration portion) | link |
| c. 1833–1871 | Analytical Engine | Charles Babbage | Store, mill, punched-card program, conditional branching | design, partial mechanism | link |
| 1838 (first use 1844) | Morse code | Samuel Morse; Alfred Vail (share disputed) | Characters as timed on/off signal sequences | electric telegraph | link |
| 1840 | Ternary calculating machine | Thomas Fowler | Balanced-ternary multiplication and division | wooden machine | link |
| 1843 | Notes (Note G) | Ada Lovelace | Program for Bernoulli numbers; the engine can act on symbols other than numbers | text | link |
| 1843–1853 | Scheutz difference engine | Georg and Edvard Scheutz | Working printing difference engine after Babbage | gear mechanism | link |

| Date | Method, system or device | Person(s) or culture | Core idea | Medium | Source |
|---|---|---|---|---|---|
| 1847 (relations 1860) | Formal Logic | Augustus De Morgan | Numerically definite syllogism, quantified predicate (disputed with Hamilton), De Morgan's laws; logic of relations 1860 | text | link |
| 1847/1854 | Boolean algebra | George Boole | Logic as an algebra over 0 and 1 (Laws of Thought) | text | link |
| 1854 | Polar planimeter | Jakob Amsler | Area integrated by tracing the boundary | brass instrument | link |
| 1861 | Recursive arithmetic | Hermann Grassmann | Addition and multiplication defined recursively from successor | text | link |
| 1867 | Lill's method | Eduard Lill | Right-angled coefficient path finds real polynomial roots | geometric construction | link |
| 1869 | Logic piano | W. Stanley Jevons | Keyboard eliminates inconsistent Boolean combinations (4 terms, 16 combinations) | wooden machine | link |
| 1872 | Tide-predicting machine | William Thomson (Lord Kelvin) | Pulleys sum harmonic constituents to predict tides | analog mechanism | link |
| 1873 | Pinwheel arithmometer | Willgodt Odhner | Compact pinwheel calculator, mass-produced from 1890 | gear mechanism | link |
| 1874 | Baudot code | Émile Baudot | Fixed 5-unit binary character code for multiplexed telegraphy | telegraph | link |
| 1876 | Ball-and-disk integrator and harmonic analyser | James Thomson; William Thomson | Mechanical integration; Fourier coefficients computed analogically | analog mechanism | link |
| 1879 | Begriffsschrift | Gottlob Frege | Formal predicate logic with quantifiers (strictly second-order, with identity) | 2-D notation, text | link |

| Date | Method, system or device | Person(s) or culture | Core idea | Medium | Source |
|---|---|---|---|---|---|
| 1880 | Venn diagrams | John Venn | Overlapping regions show all 2^n Boolean class combinations | diagrams | link |
| 1880–1886 | Switching logic, quantification, NOR/NAND | Charles S. Peirce | Single sufficient connective; logic realisable with switching circuits | letter, text | link |
| 1881 | Logic machine | Allan Marquand | Improved mechanical logic machine for four terms | wood, mechanical device | link |
| 1884 | Nomography | Maurice d'Ocagne | Alignment charts solve equations with a straightedge | printed charts | link |
| 1887 | Comptometer | Dorr E. Felt | Key-driven adding machine | key mechanism | link |
| 1888 | Printing adding machine | William S. Burroughs | Adding machine with printed record | key and print mechanism | link |
| 1888 | Recursion and arithmetic | Richard Dedekind | Natural numbers as chains; definition by recursion justified | text | link |
| 1889 | Peano axioms | Giuseppe Peano | Axioms for natural numbers with induction | text | link |
| 1889; 1893 | Direct multiplier; the Millionaire | Léon Bollée; Otto Steiger | Direct multiplication via a mechanical multiplication table | gear mechanism | link |
| 1890 | Tabulator | Herman Hollerith | Electrically sensed punched cards tabulate census data | electromechanical cards | link |
| 1890–1905 | Algebra of logic | Ernst Schröder | Systematised algebra of logic and relations | text | link |
| 1893–1895 | Algebraic machines | Leonardo Torres Quevedo | Analog machines solve polynomial equations, incl. complex roots | analog mechanism | link |
| 1895 | Runge (later Runge–Kutta) method | Carl Runge (Heun 1900, Kutta 1901) | Higher-order stepwise integration of ODEs | text | link |
| 1896 | Symbolic Logic | Lewis Carroll (C. L. Dodgson) | Diagrammatic and algorithmic method for sorites | text, board diagrams | link |

| Date | Method, system or device | Person(s) or culture | Core idea | Medium | Source |
|---|---|---|---|---|---|
| 1898 | Harmonic analyser | A. A. Michelson; S. W. Stratton | 80-component Fourier analysis and synthesis with springs | analog mechanism | link |

In the net: Fowler's machine of 1840 works in the balanced-ternary address, in which every integer has one word in digits −1, 0, +1, a carry closes as a ring (x,x,x, carry ±1) and a digit and its negative form a meeting (x,−x, carry 0). Grassmann, Dedekind and Peano rebuild arithmetic as one ordered path from a first element (N1, N3), and the punched cards of Jacquard, Babbage and Hollerith hold that path as a stored sequence. In Boole's 0 and 1 and in the logic machines a combination either closes or is struck out, while each response stays a state function of its inputs (N4). Corresponded.

## Catalogue 2 — 1900–1965

232 items accepted from 232 candidates (216 unchanged, 16 corrected), 0 rejected, judged under the respondent–judge protocol.

## Foundations, logic and computability

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1900 | Hilbert's 23 problems (incl. 10th: Diophantine algorithm) | D. Hilbert | Asks for a mechanical procedure to decide solvability of Diophantine equations | none (theory) | link |
| 1907 | Intuitionism / constructive mathematics | L.E.J. Brouwer | Mathematical existence requires construction; rejects excluded middle for infinite sets | none | link |
| 1908 | Zermelo set theory | E. Zermelo | Axiomatic set theory as formal basis for mathematics | none | link |
| 1910–1913 | Principia Mathematica / ramified type theory | B. Russell, A.N. Whitehead | Derive mathematics from logic in a typed formal system | none | link |
| 1913 | Sheffer stroke (NAND completeness) | H.M. Sheffer (anticipated by Peirce) | One binary connective suffices for all propositional logic | later basis of NAND-gate logic | link |
| 1915/1920 | Löwenheim–Skolem theorem | L. Löwenheim, T. Skolem | Countable first-order theories with infinite models have countable models | none | link |
| 1918/1932 | Strict implication / modal logic | C.I. Lewis | Formal systems for necessity and possibility (S1–S5) | none | link |
| 1920/1924 | Three-valued logic and Polish notation | J. Łukasiewicz | Third truth value; parenthesis-free prefix notation | later stack evaluation (RPN) | link |
| 1921 | Post's truth tables, completeness and many-valued logic | E. Post | Propositional calculus decidable/complete; n-valued logics | none | link |
| 1921 (publ. 1943/1965) | Post tag systems | E. Post | Delete-and-append string rewriting; later shown universal | none | link |

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1921–1928 | Hilbert's program | D. Hilbert, P. Bernays | Finitary consistency proof for all of formalized mathematics | none | link |
| 1923 | Primitive recursive arithmetic | T. Skolem | Arithmetic built from recursive definitions without quantifiers | none | link |
| 1924 | Combinatory logic | M. Schönfinkel | Eliminate bound variables using combinators S and K | later graph-reduction machines | link |
| 1928 | Ackermann function | W. Ackermann (also G. Sudan) | Total computable function not primitive recursive | none | link |
| 1928 | Entscheidungs problem | D. Hilbert, W. Ackermann | Demand for algorithm deciding validity in first-order logic | none | link |
| 1929 | Presburger arithmetic | M. Presburger | Arithmetic with addition only is decidable | none | link |
| 1929/1930 | Gödel completeness theorem | K. Gödel | Every valid first-order formula is provable | none | link |
| 1930 | Herbrand's theorem | J. Herbrand | Reduces first-order validity to propositional instances | basis of later theorem provers | link |
| 1930 | Heyting intuitionistic logic | A. Heyting | Formal axiomatization of intuitionistic logic | none | link |
| 1930 | Curry's combinatory logic | H.B. Curry | Combinators as foundation of logic and functions | none | link |
| 1931 | Gödel incompleteness theorems, Gödel numbering | K. Gödel | Consistent sufficiently strong systems are incomplete; syntax arithmetized | none | link |

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1931/1934 | Herbrand–Gödel general recursive functions | J. Herbrand, K. Gödel | Computable functions defined by systems of recursion equations | none | link |
| 1932–1951 | Recursive function theory (primitive recursion classes) | R. Péter | Systematic theory of recursive functions; nested recursion | none | link |
| 1932/1936 | Lambda calculus | A. Church | Functions as abstraction and application; computability as lambda-definability | later LISP, functional languages | link |
| 1933 | Tarski's undefinability of truth / semantic truth | A. Tarski | Truth for a language definable only in a metalanguage | none | link |
| 1934/1935 | Sequent calculus and natural deduction | G. Gentzen (also S. Jaśkowski) | Structural proof calculi; cut elimination | basis of proof search | link |
| 1934 | Curry–Howard correspondence (first form) | H.B. Curry (W. Howard 1969, publ. 1980) | Combinator types correspond to axioms of implicational logic | none | link |
| 1936 | Kleene normal form, mu-recursion | S.C. Kleene | Every recursive function = U(mu y T(e,x,y)) | none | link |
| 1936 | Church–Rosser theorem | A. Church, J.B. Rosser | Lambda reduction is confluent; normal forms unique | none | link |
| 1936 | Rosser's trick | J.B. Rosser | Incompleteness requires only consistency, not omega-consistency | none | link |
| 1936 | Church's theorem (undecidability) | A. Church | Entscheidungsproblem unsolvable | none | link |

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1936 | Turing machine, universal machine, halting problem | A.M. Turing | Tape-head-state machine defines computability; one machine simulates all | conceptual; later stored-program computers | link |
| 1936 | Post–Turing machine (Formulation 1) | E. Post | Worker moving on boxes with mark/erase instructions | none | link |
| 1936–1952 | Church–Turing thesis | A. Church, A. Turing, S. Kleene | Effective calculability equals Turing/lambda/ recursive computability | none | link |
| 1938 | Kleene recursion theorem | S.C. Kleene | Programs can obtain and use their own description | none | link |
| 1939 | Oracle machines / relative computability | A.M. Turing | Machines querying an oracle; hierarchy of unsolvability | none | link |
| 1940 | Simply typed lambda calculus | A. Church | Types on lambda terms; guarantees normalization | none | link |
| 1941 | Post's lattice (clones of Boolean functions) | E. Post | Complete classification of closed Boolean function classes | none | link |
| 1943 | Post canonical systems / production systems | E. Post | String-rewriting productions generate all recursively enumerable sets | ancestor of grammars and rule systems | link |
| 1943 | Kalmár elementary functions | L. Kalmár | Subclass of primitive recursive functions closed under bounded sums/products | none | link |
| 1943/1946 | Arithmetical hierarchy | S.C. Kleene (A. Mostowski) | Classify sets by alternating quantifier prefixes | none | link |

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1944 | Post's problem / degrees of unsolvability | E. Post | Are there r.e. degrees between computable and halting? | none | link |
| 1945 | Kleene realizability | S.C. Kleene | Intuitionistic truth interpreted via computable witnesses | none | link |
| 1946 | Post correspondence problem | E. Post | Simple tile-matching problem shown undecidable | none | link |
| 1947 | Word problem for semigroups undecidable | E. Post, A.A. Markov Jr. | First undecidable problem from classical algebra | none | link |
| 1948/1951 | Tarski decision procedure for real closed fields | A. Tarski | Elementary algebra and geometry of reals is decidable | none | link |
| 1950 | Trakhtenbrot's theorem | B. Trakhtenbrot | Validity over finite models is not decidable | none | link |
| 1951/1954 | Markov normal algorithms | A.A. Markov Jr. | Ordered string substitution rules as definition of algorithm | later Refal, text rewriting | link |
| 1953 | Rice's theorem | H.G. Rice | All nontrivial semantic properties of programs are undecidable | none | link |
| 1953 | Grzegorczyk hierarchy | A. Grzegorczyk | Stratifies primitive recursive functions by growth rate | none | link |
| 1953/1958 | Kolmogorov–Uspensky machine | A.N. Kolmogorov, V.A. Uspensky | Computation on a changing graph with local active zone | none | link |
| 1955 | Beth semantic tableaux | E.W. Beth (also Hintikka) | Refutation trees for deciding/proving formulas | basis of tableau provers | link |

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1955/1958 | Novikov–Boone theorem (word problem for groups) | P. Novikov, W. Boone | Finitely presented groups with undecidable word problem | none | link |
| 1956/1957 | Friedberg–Muchnik theorem / priority method | R. Friedberg, A. Muchnik | Solves Post's problem: incomparable r.e. degrees exist | none | link |
| 1957 | Craig interpolation | W. Craig | Implication between formulas has interpolant in shared vocabulary | later model checking | link |
| 1958 | Gödel Dialectica interpretation | K. Gödel | Arithmetic interpreted by computable functionals of finite type (System T) | none | link |
| 1958 | Lambek calculus | J. Lambek | Substructural type logic for sentence structure | categorial parsers | link |
| 1958 | Yanov program schemata | Yu. Yanov | Abstract program schemes with decidable equivalence | none | link |
| 1961 | Wang tiles / domino problem | Hao Wang | Tiling the plane with colored squares encodes computation | none | link |
| 1961 | Davis–Putnam–Robinson theorem | M. Davis, H. Putnam, J. Robinson | Exponential Diophantine sets = r.e. sets (toward Hilbert 10) | none | link |
| 1961 | Minsky counter machine / tag universality | M. Minsky | Two counters suffice for universality; tag systems universal | none | link |
| 1961/1963 | Basis for a mathematical theory of computation | J. McCarthy | Recursive definitions, conditional expressions, abstract syntax | none | link |

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1963 | Shepherdson–Sturgis register machine | J. Shepherdson, H. Sturgis | Unlimited registers with increment/decrement/ jump compute all recursive functions | model of real computers | link |
| 1963 | Cohen forcing / independence of CH | P. Cohen | Continuum hypothesis independent of ZFC | none | link |
| 1964 | Random-access stored-program machine models | C. Elgot, A. Robinson | Abstract RASP machines with program in memory | none | link |

In the net: A formal proof is a strand: one ordered path of steps (N1) that ends when a window of three trits sums to 0 mod 3, so a proof ends, a run halts or a fixed point is reached as closure (N2). Gödel numbering and the balanced-ternary address express the same structure, every formula and every integer having exactly one word; what a system does not close within itself (incompleteness, undecidability, undefinability of truth) passes as carry to the next scale, the metalanguage or the oracle (N5). Corresponded.

**Automata, formal languages and complexity**

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1943 | McCulloch–Pitts neuron | W. McCulloch, W. Pitts | Logical threshold neurons; nets compute Boolean/finite-state functions | none (theory) | link |
| 1948–1952 (publ. 1966) | Von Neumann cellular automata / self-reproduction | J. von Neumann, S. Ulam | 29-state cellular automaton contains universal constructor | none | link |
| 1951/1956 | Kleene regular expressions and theorem | S.C. Kleene | Events of nerve nets/ finite automata = regular expressions | later grep, lexers | link |
| 1955 | Mealy machine | G.H. Mealy | Finite-state machine with outputs on transitions | sequential circuit design | link |
| 1956 | Moore machine / Gedanken-experiments | E.F. Moore | Outputs depend on state; experimental identification of machines | sequential circuits | link |
| 1956 | Automata Studies | C. Shannon, J. McCarthy (eds.) | Founding collection of automata theory | none | link |
| 1956 | Shannon's two-state universal Turing machine | C. Shannon | Universal TM possible with only two states (or two symbols) | none | link |
| 1956/1959 | Chomsky hierarchy, generative grammar | N. Chomsky | Type 0–3 grammars correspond to automata classes | parsers, compilers | link |
| 1957/1958 | Myhill–Nerode theorem | J. Myhill, A. Nerode | Regularity iff finite index of right-congruence; minimization | none | link |
| 1959 | Rabin–Scott nondeterministic automata | M. Rabin, D. Scott | NFA equals DFA via subset construction | none | link |
| 1960–1962 | Büchi–Elgot–Trakhtenbrot theorem | J.R. Büchi, C. Elgot, B. Trakhtenbrot | Regular languages = monadic second-order definable | later model checking | link |

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1960 | Linear bounded automata | J. Myhill (P. Landweber 1963, S.-Y. Kuroda 1964) | Turing machine bounded by input length; context-sensitive languages | none | link |
| 1960 | Rabin degrees of difficulty of computation | M. Rabin | Abstract measure of computation difficulty; hierarchies exist | none | link |
| 1961–1963 | Pushdown automata | A. Oettinger, M.-P. Schützenberger, N. Chomsky | Finite control plus stack recognizes context-free languages | parsers | link |
| 1961 | Bar-Hillel pumping lemma | Y. Bar-Hillel, M. Perles, E. Shamir | Structural limits of context-free languages | none | link |
| 1961 | Glushkov abstract automata theory, Glushkov construction | V.M. Glushkov | Algebraic synthesis of automata from regular expressions | Kiev computer design | link |
| 1961–1970 | CYK algorithm | I. Sakai (1961/62); T. Kasami (1965); D. Younger (1967); J. Cocke, J.T. Schwartz (1970) | Cubic-time parsing for context-free grammars | none | link |
| 1962 | Büchi automata (infinite words) | J.R. Büchi | Automata on infinite sequences; decidability of S1S | later verification | link |
| 1962 | Yamada real-time computation | H. Yamada | Classes of functions computable in real time | none | link |
| 1962 | Petri nets | C.A. Petri | Places, transitions, tokens model concurrent asynchronous systems | workflow/ protocol modelling | link |

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1963 | Ritchie hierarchy (space-bounded function classes) | R.W. Ritchie | Space-bounded TM classes; union = Kalmár elementary functions; lowest level = linear space | none | link |
| 1964/1965 | Cobham–Edmonds thesis | A. Cobham, J. Edmonds | Feasible computation equals polynomial time | none | link |
| 1965 | Hartmanis–Stearns time hierarchy | J. Hartmanis, R. Stearns | Computational complexity classes by time; more time computes more | none | link |
| 1965 | Space complexity hierarchy | R. Stearns, J. Hartmanis, P. Lewis | Memory-bounded classes and space hierarchy | none | link |
| 1965 | Knuth LR parsing | D. Knuth | Deterministic bottom-up parsing of LR(k) grammars | compiler-compilers | link |

In the net: An automaton is a context function: its response is a function of the current state (N4), and the first symbols read fix the direction against which later symbols are read (N3). Finite automata close within a bounded window; the stack, the linear bound and the unbounded tape are successive scales that hold what has not yet closed as carry (N5), and the complexity hierarchies give the number of passages needed before closure. Corresponded.

## Information, coding and communication

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1915 | Whittaker cardinal interpolation | E.T. Whittaker | Band-limited functions reconstructed from samples | none | link |
| 1924/1928 | Nyquist telegraph speed / sampling | H. Nyquist | Signalling rate bounded by twice bandwidth | telegraphy | link |
| 1925 | Fisher information | R.A. Fisher | Information a sample carries about a parameter | none | link |
| 1928 | Hartley information | R.V.L. Hartley | Information = log of number of possible messages | telephony | link |
| 1929 | Szilard engine | L. Szilard | Measurement/ information linked to thermodynamic entropy | thought experiment | link |
| 1933 | Kotelnikov sampling theorem | V. Kotelnikov | Independent sampling theorem for communication | radio | link |
| 1946 | Gabor information cells | D. Gabor | Time-frequency logons as quanta of information | none | link |
| 1948 | Shannon mathematical theory of communicatio | C.E. Shannon | Entropy, channel capacity, coding theorems | basis of digital communicatio n | link |
| 1948/1949 | Shannon–Fano coding | C. Shannon, R. Fano | Prefix code from probability splitting | compression | link |
| 1949 | Shannon secrecy systems | C.E. Shannon | Information-theoretic cryptography; one-time pad perfect secrecy | cryptographic design | link |
| 1949 | Golay code | M. Golay | Perfect multi-error-correcting binary code | Voyager (later) | link |

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1950 | Hamming codes and distance | R. Hamming | Single-error-correcting codes using parity checks | motivated by errors on Bell Labs Model V relay computer; not implemented there | link |
| 1952 | Huffman coding | D.A. Huffman | Optimal prefix code built bottom-up | compression formats | link |
| 1953/1956 | Brillouin negentropy principle | L. Brillouin | Information acquisition costs physical entropy | none | link |
| 1953/1956 | Khinchin axiomatic entropy | A.Ya. Khinchin | Rigorous axiomatization of Shannon theory | none | link |
| 1954 | Reed–Muller codes | I. Reed, D. Muller | Codes from multivariate Boolean polynomials | Mariner 9 (later) | link |
| 1955 | Convolutional codes | P. Elias | Codes generated by sliding shift-register | deep-space links | link |
| 1959/1960 | BCH codes | A. Hocquenghem, R.C. Bose, D. Ray-Chaudhuri | Cyclic codes with designed multi-error correction | storage, communication | link |
| 1960 | Reed–Solomon codes | I. Reed, G. Solomon | Polynomial codes over finite fields | CDs, QR (later) | link |
| 1960/1963 | LDPC codes | R. Gallager | Sparse parity-check codes with iterative decoding | rediscovered 1990s | link |
| 1960/1964 | Algorithmic probability / inductive inference | R. Solomonoff | Prediction via universal prior over programs | none | link |
| 1961 | Cyclic redundancy check | W.W. Peterson, D.T. Brown | Polynomial division for error detection | networks, disks | link |
| 1961 | Landauer's principle | R. Landauer | Erasing one bit dissipates at least kT ln 2 | none | link |

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1962 | Bremermann's limit | H. Bremermann | Maximum computation rate per mass from physics | none | link |
| 1965 | Kolmogorov complexity | A.N. Kolmogorov (G. Chaitin 1966) | Information of an object = length of shortest program | none | link |

In the net: A code word is a window that closes: the parity check is a meeting closure (x, −x, carry 0), and a detected error is a window that stays open (N2). Sampling and entropy give the number of trits a strand needs before it closes, and Landauer's bound states that erasing an open window passes energy as carry to the thermal scale (N5). Corresponded.

**Control, cybernetics and systems**

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1922 | PID control theory | N. Minorsky | Proportional-integral-derivative automatic ship steering | ship autopilots | link |
| 1927 | Negative feedback amplifier | H.S. Black | Feedback stabilizes gain against component variation | telephone repeaters | link |
| 1928/1944 | Von Neumann theory of games | J. von Neumann (with O. Morgenstern 1944) | Minimax strategies for decision under conflict | none | link |
| 1932 | Nyquist stability criterion | H. Nyquist | Closed-loop stability from open-loop frequency plot | amplifiers, servos | link |
| 1939 | Linear programming (USSR) | L. Kantorovich | Optimal resource allocation as linear optimization | planning | link |
| 1940/1945 | Bode feedback theory | H.W. Bode | Gain-phase relations and limits of feedback design | servos | link |
| 1941/1942 (publ. 1949) | Kolmogorov–Wiener filtering | A.N. Kolmogorov, N. Wiener | Optimal linear prediction of stationary processes | anti-aircraft predictors | link |
| 1943 | Behavior, Purpose and Teleology | A. Rosenblueth, N. Wiener, J. Bigelow | Purposeful behaviour explained by negative feedback | none | link |
| 1946–1953 | Macy conferences | W. McCulloch, M. Mead, G. Bateson, von Neumann et al. | Interdisciplinary founding of cybernetics and systems thinking | none | link |
| 1947 | Simplex method | G. Dantzig | Pivoting along vertices solves linear programs | early computers (SEAC) | link |
| 1948 | Cybernetics | N. Wiener | Control and communication in animal and machine | none | link |

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1948/1952 | Homeostat / ultrastability | W.R. Ashby | Adaptive system reaching stability via random step changes | Homeostat | link |
| 1948–1949 | Grey Walter tortoises | W. Grey Walter | Simple analog circuits produce complex autonomous behaviour | Elmer and Elsie robots | link |
| 1953/1957 | Dynamic programming / Bellman equation | R. Bellman | Optimal decisions via recursive value function | none | link |
| 1953/1955 | Lyapunov operator method / Soviet cybernetics | A.A. Lyapunov, S.L. Sobolev, A.I. Kitov | Programs as operator schemes; cybernetics rehabilitated in USSR | Strela, BESM programming | link |
| 1953 | Musicolour / conversation-based learning machines | G. Pask | Adaptive machine responding to performer | Musicolour | link |
| 1956 | Law of requisite variety | W.R. Ashby | Only variety can absorb variety in regulation | none | link |
| 1956/1962 | Pontryagin maximum principle | L. Pontryagin et al. | Necessary conditions for optimal control | aerospace | link |
| 1956/1959 | Kitov "Electronic Digital Machines" and EASU network plan | A.I. Kitov | Computer theory text; nationwide computer network for planning | proposal (rejected) | link |
| 1959/1960 | Stratonovich nonlinear filtering | R. Stratonovich | Optimal estimation of nonlinear stochastic processes | none | link |
| 1959 | Cybernetic management (later Viable System Model) | S. Beer | Organization as recursive cybernetic control system | none | link |

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1960 | Kalman filter, state space, controllability | R.E. Kalman | Recursive optimal state estimation; controllability/ observability | Apollo navigation | link |
| 1960 | Self-organizing systems | H. von Foerster | Order from noise; self-organizing systems need an environment | BCL Illinois | link |
| 1962 | OGAS | V.M. Glushkov | National automated economic management network | not realized | link |
| 1962 | Architecture of complexity | H.A. Simon | Near-decomposable hierarchies make complex systems evolvable | none | link |
| 1965 | Fuzzy sets | L. Zadeh | Graded membership instead of binary | later fuzzy control | link |

In the net: Negative feedback is a meeting closure: a deviation x meets −x, the carry is 0 and the loop returns to standing 0 (N2). Stability criteria, filters and optimal control give the conditions under which this closure holds along the strand, with the response a state function (N4); what does not close passes as carry to the next level of regulation, the hierarchy described by Ashby, Beer and Simon (N5). Corresponded.

## Switching theory and logic design

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1934–1938 | Switching circuit theory (Japan) | A. Nakashima, M. Hanzawa | Algebraic theory of relay networks | relay circuits | link |
| 1935/1941 | Relay-contact logic (USSR) | V. Shestakov | Boolean algebra for relay circuits, independent of Shannon | relay circuits | link |
| 1937/1938 | Symbolic analysis of relay circuits | C.E. Shannon | Boolean algebra designs and simplifies switching circuits | all digital logic | link |
| 1952/1956 | Quine–McCluskey minimization | W.V. Quine, E. McCluskey | Tabular minimization of Boolean functions | logic design tools | link |
| 1952/1953 | Veitch chart / Karnaugh map | E. Veitch, M. Karnaugh | Graphical Boolean simplification | logic design | link |
| 1954 | Huffman sequential switching synthesis | D.A. Huffman | Systematic design of asynchronous sequential circuits | relay/ electronic circuits | link |
| 1956 | Reliable computation from unreliable components | J. von Neumann; E. Moore, C. Shannon | Redundancy (multiplexing/majority) yields arbitrarily reliable circuits | fault-tolerant design | link |

In the net: A switching circuit reads a window of inputs and closes it to one output; after Boolean minimization fewer windows give the same closure (N2). Sequential circuits add memory, the first passage fixing the state against which later inputs are read (N3), and three equal majority readings merge into one reliable unit as a ring closure (x, x, x) at the next scale (N5). Corresponded.

## Machines and architecture

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1909 | Ludgate analytical machine design | P. Ludgate | Program-controlled mechanical computer with logarithmic multiplication | design only | link |
| 1914 | Essays on Automatics | L. Torres Quevedo | Electromechanical automata; floating-point notation proposed | El Ajedrecista, Electromechanical Arithmometer 1920 | link |
| 1928–1931 | Differential analyzer | V. Bush, H.L. Hazen | Mechanical integration of differential equations | MIT differential analyzer | link |
| 1937/1940 | Stibitz relay calculators, remote computing | G. Stibitz | Binary relay arithmetic; first remote operation via teletype | Model K, Complex Number Calculator | link |
| 1937–1942 | Atanasoff–Berry Computer | J. Atanasoff, C. Berry | Electronic binary arithmetic with regenerative capacitor memory | ABC | link |
| 1938 | Binary calculating machine proposal | L. Couffignal | Argued for binary arithmetic in calculating machines | design | link |
| 1938–1945 | Zuse binary program-controlled computers | K. Zuse | Binary floating-point, punched-tape program control | Z1, Z3 (1941), Z4 | link |
| 1941 | General purpose analog computer theory | C.E. Shannon | Mathematical theory of differential analyzer computable functions | differential analyzers | link |
| 1942–1945 | Plankalkül | K. Zuse | First high-level programming language design | implemented 1975/2000 | link |
| 1943–1944 | Colossus | T. Flowers, M. Newman | Programmable electronic valve machine for cryptanalysis | Colossus Mk1/ Mk2 | link |

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1944 | Separate instruction and data storage (term "Harvard architecture" coined later) | H. Aiken | Instructions on punched tape and data in counters kept separate | Harvard Mark I (ASCC) | link |
| 1945/1946 | ENIAC / Eckert–Mauchly designs | J.P. Eckert, J. Mauchly | General-purpose electronic digital computer; later UNIVAC | ENIAC, BINAC, UNIVAC I | link |
| 1945 | Von Neumann architecture (EDVAC report) | J. von Neumann (with Eckert, Mauchly) | Stored program in shared memory; sequential CPU | EDVAC, IAS machine | link |
| 1945/1946 | Turing ACE design | A.M. Turing | Detailed stored-program design with subroutines | Pilot ACE (1950) | link |
| 1945 | Memex | V. Bush | Associative trails linking stored documents | concept | link |
| 1947 | Williams tube memory | F. Williams, T. Kilburn | Random-access storage on CRT | Manchester Baby (1948) | link |
| 1947 | Assembly language | K. Booth | Symbolic notation for machine instructions | ARC | link |
| 1947 | Transistor | W. Shockley, J. Bardeen, W. Brattain | Solid-state switching/amplification | later transistor computers | link |
| 1949/1951 | EDSAC and subroutine library (Wheeler jump) | M. Wilkes, D. Wheeler, S. Gill | Practical stored-program service; closed subroutines | EDSAC | link |
| 1949–1953 | Magnetic-core memory | J. Forrester, An Wang, J. Rajchman | Coincident-current ferrite core random-access memory | Whirlwind | link |
| 1949 | Index registers (B-lines) | T. Kilburn, F. Williams | Address modification register for loops | Manchester Mark 1 | link |

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1951 | Microprogramming | M. Wilkes | Control unit as stored micro-instruction sequences | EDSAC 2 (1958) | link |
| 1951/1952 | MESM / BESM | S.A. Lebedev | First Soviet stored-program computers | MESM, BESM | link |
| 1951/1954 | Böhm compiler in its own language | C. Böhm | Compiler described in the language it compiles | design | link |
| 1952 | Compiler / A-0 system | G. Hopper | Automatic translation of symbolic routines to machine code | UNIVAC I | link |
| 1952 | Autocode | A. Glennie | Early compiled high-level language | Manchester Mark 1 | link |
| 1955/1957 | Stack principle (Kellerprinzip) | F.L. Bauer, K. Samelson | Expression translation using a push-down store | ALGOL compilers | link |
| 1957 | FORTRAN | J. Backus et al. (IBM) | Optimizing compiler for formula translation | IBM 704 | link |
| 1958/1960 | ALGOL 58/60 | ACM-GAMM committee (1958); ACM/ European committee, ed. P. Naur (1960) | Block structure, recursion, formally defined syntax | many compilers | link |
| 1958/1960 | LISP | J. McCarthy | Recursive functions of symbolic expressions; eval; garbage collection | IBM 704 | link |
| 1958 | Ershov programming program and optimization | A.P. Ershov | Early Soviet translator; common subexpression elimination | BESM | link |
| 1958 | Setun ternary computer | N. Brusentsov, S. Sobolev | Balanced ternary arithmetic in production computer | Setun | link |

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1959/1960 | Backus–Naur Form | J. Backus, P. Naur | Formal metasyntax for programming-language grammars | ALGOL 60 report | link |
| 1959 | COBOL | CODASYL, G. Hopper | Business-oriented English-like language | many mainframes | link |
| 1959/1961 | Time-sharing | C. Strachey, J. McCarthy, F. Corbató | Multiple users share computer interactively | CTSS | link |
| 1960 | Man-computer symbiosis | J.C.R. Licklider | Interactive human-computer partnership | ARPA IPTO | link |
| 1961 | Stretch pipelining and lookahead | IBM (W. Buchholz, F. Brooks et al.) | Instruction overlap, byte concept, interrupts | IBM 7030 | link |
| 1961 | Stack machine architecture | R. Barton (Burroughs) | Hardware stack for high-level language execution | Burroughs B5000 | link |
| 1962 | Virtual memory / paging | T. Kilburn et al. | One-level store with automatic page transfer | Atlas | link |
| 1962 | APL notation | K. Iverson | Array-based mathematical notation for programs | APL\360 (1966) | link |
| 1962 | Augmenting human intellect | D. Engelbart | Conceptual framework for intellect-amplifying systems | NLS (1968), mouse | link |
| 1963 | Sketchpad | I. Sutherland | Interactive graphics with constraints and object instances | TX-2 | link |
| 1964 | Compatible architecture family | G. Amdahl, G. Blaauw, F. Brooks | Architecture separated from implementation across models | IBM System/ 360 | link |
| 1964 | Multiple functional units / scoreboarding | S. Cray, J. Thornton | Parallel functional units with dynamic scheduling | CDC 6600 | link |

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1964 | SECD machine | P. Landin | Abstract machine evaluating lambda expressions | interpreters | link |
| 1965 (development) | MIR computers (Machine for Engineering Calculations) | V.M. Glushkov | High-level language interpreted in hardware | MIR-1 | link |
| 1965 | Mutual exclusion and semaphores | E.W. Dijkstra | Synchronization primitives for cooperating processes | THE system | link |

In the net: A stored-program machine runs one strand of instructions in order (N1) and the run ends when it closes (N2); a counter overflow is a ring closure (x, x, x, carry ±1) into the next digit, and in memory hierarchies and paging what does not fit at one scale passes as carry to the next (N5). The Setun computer used the balanced-ternary address directly: every integer has one word in the digits −1, 0, +1. Corresponded.

## Algorithms

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1926 | Borůvka minimum spanning tree | O. Borůvka | Greedy component merging for minimal network | electric grid design | link |
| 1926 | Grete Hermann effective ideal algorithms | G. Hermann | Finite procedures for polynomial ideal questions | none | link |
| 1945 | Merge sort | J. von Neumann | Divide-and-conquer sorting by merging | EDVAC | link |
| 1946/1949 | Monte Carlo method | S. Ulam, J. von Neumann, N. Metropolis | Random sampling to compute numerical results | ENIAC | link |
| 1953 | Metropolis algorithm | N. Metropolis et al. | Markov-chain sampling of distributions | MANIAC | link |
| 1955 | Hungarian algorithm | H. Kuhn | Polynomial assignment problem solution | none | link |
| 1956 | Max-flow min-cut / Ford–Fulkerson | L. Ford, D. Fulkerson | Augmenting paths compute maximum network flow | none | link |
| 1956/1957 | Kruskal and Prim spanning trees | J. Kruskal; R. Prim | Greedy minimum spanning tree algorithms | none | link |
| 1956–1958 | Bellman–Ford shortest paths | R. Bellman, L. Ford | Shortest paths with negative weights by relaxation | routing | link |
| 1959 | Dijkstra shortest path | E.W. Dijkstra | Greedy single-source shortest paths | ARMAC demo | link |
| 1959–1961 | QR algorithm | J. Francis, V. Kublanovskaya | Iterative eigenvalue computation by QR factorization | numerical libraries | link |
| 1959/1961 | Quicksort | C.A.R. Hoare | Partition-based recursive sorting | ALGOL 60 procedure (CACM Algorithm 64) | link |
| 1960 | Karatsuba multiplication | A. Karatsuba | Multiplication faster than quadratic; refuted Kolmogorov conjecture | none | link |

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1962 | AVL tree | G. Adelson-Velsky, E. Landis | Self-balancing binary search tree | none | link |
| 1962 | Gale–Shapley stable matching | D. Gale, L. Shapley | Deferred acceptance yields stable matching | none | link |
| 1964 | Heapsort | J.W.J. Williams | Sorting via binary heap | none | link |
| 1965 | Edmonds blossom algorithm | J. Edmonds | Polynomial maximum matching in general graphs | none | link |
| 1965 | Fast Fourier transform | J. Cooley, J. Tukey (Gauss, Runge, Danielson–Lanczos earlier) | O(n log n) discrete Fourier transform | IBM 7094 | link |
| 1965 | Levenshtein distance | V. Levenshtein | Edit distance between strings | none | link |
| 1965 | Buchberger's algorithm | B. Buchberger | Gröbner bases for polynomial ideals | computer algebra | link |

In the net: An algorithm is a strand of steps that closes at a fixed point: a sorted list, a shortest path, a maximum flow, a converged eigenvalue (N1, N2). In divide-and-conquer methods the strand splits into windows that close locally, and the remainder passes as carry to the merge level (N5); in greedy methods each window closes at the first passage and later choices are read relative to it (N3). Corresponded.

## Automated reasoning and early AI

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1948 | Turing's unorganized machines | A.M. Turing | Randomly connected NAND networks trainable toward organization | none | link |
| 1949 | Hebbian learning | D. Hebb | Co-active neurons strengthen connections | later neural nets | link |
| 1950 | Turing test / Computing Machinery and Intelligence | A.M. Turing | Imitation game as operational criterion for machine thinking | none | link |
| 1950 | Shannon chess program | C.E. Shannon | Minimax search with evaluation for game playing | none | link |
| 1951 | SNARC | M. Minsky, D. Edmonds | Random-wired reinforcement neural net machine | SNARC (vacuum tubes) | link |
| 1955/1956 | Dartmouth proposal (artificial intelligence) | J. McCarthy, M. Minsky, N. Rochester, C. Shannon | Every feature of intelligence can be simulated by machine | none | link |
| 1956 | Logic Theorist and IPL | A. Newell, H. Simon, C. Shaw | Heuristic search proves Principia theorems; list processing | JOHNNIAC | link |
| 1957/1959 | General Problem Solver | A. Newell, H. Simon, C. Shaw | Means-ends analysis as universal problem-solving method | IPL program (RAND) | link |
| 1957/1958 | Perceptron | F. Rosenblatt | Trainable linear threshold classifier with convergence | Mark I Perceptron | link |
| 1958/1960 | Wang mechanical theorem proving | Hao Wang | Proved all Principia propositional/predicate theorems mechanically | IBM 704 | link |
| 1959 | Pandemonium | O. Selfridge | Parallel feature demons for pattern recognition | simulation | link |
| 1959 | Samuel checkers learning | A. Samuel | Machine learning via self-play and evaluation tuning | IBM 701/704 | link |

| Year | Theory | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1960 | ADALINE / LMS rule | B. Widrow, T. Hoff | Adaptive linear element trained by least mean squares | ADALINE (memistor) | link |
| 1960/1962 | Davis–Putnam / DPLL | M. Davis, H. Putnam; G. Logemann, D. Loveland | Satisfiability by propagation and backtracking | IBM 704 | link |
| 1962 | Adaptive systems / genetic plans | J. Holland | Logical theory of adaptive systems (precursor genetic algorithms) | none | link |
| 1964 | Evolutionary programming | L. Fogel | Evolving finite-state machines by mutation and selection | simulation | link |
| 1965 | Resolution and unification | J.A. Robinson | Single inference rule for first-order theorem proving | theorem provers, later Prolog | link |

In the net: Search and resolution run a strand of inferences until a window closes: a proof found, a clause set refuted, a game value fixed (N2). In learning machines the weights change until the response, a state function (N4), closes on the training cases; error that does not close passes as carry to the next update or generation (N5). Corresponded.

## Catalogue 3 — Theoretical computer science, 1965–2026

219 theories accepted from 224 candidates, 28 of them with corrected fields, 0 rejected, 5 already in Catalogue 2 (Hartmanis–Stearns hierarchy, Kolmogorov complexity, Curry–Howard correspondence, Petri nets, Kolmogorov–Uspensky machine); every entry passed the respondent–judge protocol, and the source column gives the judge's verification page.

## Complexity

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1967 | Blum axioms | M. Blum | machine-independent complexity measures; speedup theorem | — | link |
| 1969–2024 | Fast matrix multiplication | Strassen; Coppersmith–Winograd; Alman, Duan, Vassilevska Williams, Xu, Xu, Zhou | $\omega < 2.372$ | Strassen variant in some libraries (BLAS: classical $O(n^3)$) | link |
| 1970 | Savitch's theorem | W. Savitch | $\mathrm{NSPACE}(s) \subseteq \mathrm{DSPACE}(s^2)$ | — | link |
| 1971/1973 | Cook–Levin theorem | S. Cook; L. Levin | SAT is NP-complete | SAT solvers | link |
| 1972 | Karp's 21 NP-complete problems | R. Karp | polynomial reductions spread NP-completeness | — | link |
| 1972 | Polynomial hierarchy | A. Meyer, L. Stockmeyer | alternating quantifiers give a tower of classes | — | link |
| 1974; 1982–1988 | Descriptive complexity | R. Fagin; N. Immerman | NP = existential second-order logic | — | link |
| 1975 | Relativization | T. Baker, J. Gill, R. Solovay | oracle results: P vs NP not settled by relativizing proofs | — | link |
| 1975 | Ladner's theorem | R. Ladner | if P≠NP, NP-intermediate problems exist | — | link |
| 1976/1981 | Alternating Turing machines | A. Chandra, D. Kozen, L. Stockmeyer | existential/universal states; APTIME = PSPACE | — | link |
| 1979 | #P and the permanent | L. Valiant | counting class #P; permanent #P-complete | — | link |
| 1979 | VP vs VNP | L. Valiant | algebraic P vs NP | — | link |
| 1979 | Communication complexity | A. Yao | bits two parties must exchange | — | link |
| 1979 | Proof complexity | S. Cook, R. Reckhow | length of propositional proofs | — | link |

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1981–1986 | AC0 lower bounds (parity) | Furst–Saxe–Sipser; Ajtai; Håstad | parity not in AC0; switching lemma | — | link |
| 1985 | Monotone lower bounds | A. Razborov | clique needs superpolynomial monotone circuits | — | link |
| 1985 | Interactive proofs | Goldwasser–Micali–Rackoff; Babai | randomized verifier with all-powerful prover | — | link |
| 1987 | Immerman–Szelepcsényi | N. Immerman; R. Szelepcsényi | NL = coNL | — | link |
| 1990 | IP = PSPACE | A. Shamir; Lund–Fortnow–Karloff–Nisan | interactive proofs = polynomial space | — | link |
| 1991 | MIP = NEXP | L. Babai, L. Fortnow, C. Lund | two provers verify NEXP | — | link |
| 1991 | Toda's theorem | S. Toda | PH ⊆ P^#P | — | link |
| 1992 | PCP theorem | Arora–Safra; Arora–Lund–Motwani–Sudan–Szegedy | proofs checkable with O(1) queries | — | link |
| 1992–1999 | Parameterized complexity | R. Downey, M. Fellows | FPT and W-hierarchy | PACE challenge solvers | link |
| 1994 | Natural proofs barrier | A. Razborov, S. Rudich | natural proofs would break pseudorandom functions | — | link |
| 1997/2001 | Optimal inapproximability | J. Håstad | tight bounds for Max-E3SAT, Max-E3LIN-2, Set Splitting (clique bound: separate paper 1996/1999) | — | link |
| 1999/2001 | Exponential Time Hypothesis | R. Impagliazzo, R. Paturi | 3-SAT needs 2^Ω(n) | — | link |
| 2001 | Geometric complexity theory | K. Mulmuley, M. Sohoni | lower bounds via algebraic geometry | — | link |

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 2002 | Unique Games Conjecture | S. Khot | unique label cover hard to approximate | — | link |
| 2002 | AKS primality | M. Agrawal, N. Kayal, N. Saxena | deterministic polynomial primality | — | link |
| 2005 | Dinur's PCP proof | I. Dinur | PCP via gap amplification | — | link |
| 2005 | SL = L | O. Reingold | undirected connectivity in log space | — | link |
| 2008 | Algebrization barrier | S. Aaronson, A. Wigderson | algebrizing techniques cannot resolve P vs NP | — | link |
| 2010s | Fine-grained complexity | V. Vassilevska Williams, R. Williams | exact exponents via conditional reductions | — | link |
| 2011 | ACC0 lower bound | R. Williams | NEXP not in ACC0 | — | link |
| 2014 | Catalytic computing | Buhrman, Cleve, Koucký, Loff, Speelman | full memory usable if restored | — | link |
| 2015 | Graph isomorphism quasipolynomial | L. Babai | GI in quasipolynomial time | — | link |
| 2017 | CSP dichotomy theorem | A. Bulatov; D. Zhuk | every finite CSP is in P or NP-complete | — | link |
| 2023 | Tree evaluation in small space | J. Cook, I. Mertz | catalytic techniques, O(log n log log n) space | — | link |
| 2025 | Time–space simulation | R. Williams | TIME[t] ⊆ SPACE[√(t log t)] | — | link |

In the net: A complexity class is a bound on how many trit windows along one strand may stay open before closure (N1, N2); a certificate is a meeting (x,−x) at the verifier, and a hierarchy result states that more windows give closures unreachable with fewer. Under Theorem C1 closure with carry keeps at most two open passages per scale, so its open state grows as $\log_3 n$, the same order as the logarithmic-space classes (Reingold 2005, Cook–Mertz 2023); no derivation of those results from C1 is claimed. Corresponded.

## Algorithmic information

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1966 | Martin-Löf randomness | P. Martin-Löf | random = passes all effective tests | — | link |
| 1973 | Universal search | L. Levin | interleave programs by length and time | — | link |
| 1975 | Chaitin's Omega | G. Chaitin | halting probability, uncomputable | — | link |
| 1988 | Logical depth | C. Bennett | organization = runtime of near-shortest program | — | link |
| 2000/2018 | MCSP / meta-complexity | Kabanets–Cai; Hirahara | complexity of computing complexity | — | link |
| 2020 | One-way functions ↔ K^t | Y. Liu, R. Pass | OWF iff K^t (time-bounded Kolmogorov complexity) mildly hard on average | — | link |

In the net: Description length is the length of the shortest strand that ends in closure on a given string; what does not compress stays open as carry at every scale (N5). A random sequence is one in which no window closes before its full length, and logical depth is the number of passages between the short description and that closure. Corresponded.

## Randomized computation

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1973–2000 | Expanders and extractors | Pinsker; Margulis; Nisan–Zuckerman; Reingold–Vadhan–Wigderson | sparse well-connected graphs; extractors turn weak random sources into near-uniform bits; zig-zag product | — | link |
| 1976/1980 | Miller–Rabin test | G. Miller; M. Rabin | probabilistic primality | OpenSSL | link |
| 1977 | Solovay–Strassen test | R. Solovay, V. Strassen | randomized primality | superseded by Miller–Rabin, Baillie–PSW | link |
| 1977 | Probabilistic TM / BPP | J. Gill | bounded-error random classes | — | link |
| 1978–1980 | Schwartz–Zippel | Schwartz; Zippel; DeMillo–Lipton | random evaluation detects nonzero polynomials | — | link |
| 1982–1997 | Hardness vs randomness | Blum–Micali; Yao; Nisan–Wigderson; Impagliazzo–Wigderson | hard functions give generators | — | link |
| 1989/2001 | MCMC approximate counting | Jerrum, Sinclair, Vigoda | FPRAS for the permanent | — | link |

In the net: A random bit is a trit window read without memory of earlier passages (N3); in amplification and derandomization repeated windows pass up as one ring (x,x,x) whose carry is the majority answer. In an expander every cut is crossed by many edges, so no region of the strand stays isolated and open passages spread instead of accumulating. Corresponded.

## Cryptography as computation

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1976 | Diffie–Hellman | W. Diffie, M. Hellman (Merkle) | public-key agreement | TLS | link |
| 1977 | RSA | Rivest, Shamir, Adleman | trapdoor from factoring | TLS | link |
| 1982 | Semantic security | S. Goldwasser, S. Micali | probabilistic encryption | — | link |
| 1982/1986 | Garbled circuits | A. Yao | secure two-party computation | MPC | link |
| 1985 | Zero-knowledge proofs | Goldwasser, Micali, Rackoff | prove without revealing | zk-SNARKs | link |
| 1987 | GMW multiparty computation | Goldreich, Micali, Wigderson | any function securely computable | MP-SPDZ | link |
| 1992/1994/2013 | SNARKs | Kilian; Micali; GGPR | succinct proofs of computation | zk-rollups | link |
| 1996/2005 | Lattice crypto / LWE | M. Ajtai; O. Regev | worst-case lattice hardness | ML-KEM | link |
| 2001 | Universal composability | R. Canetti | security under composition | — | link |
| 2006 | Differential privacy | Dwork, McSherry, Nissim, Smith | outputs stable under one record | US Census 2020 | link |
| 2008 | Nakamoto consensus | S. Nakamoto | proof-of-work longest chain | Bitcoin | link |
| 2009 | Fully homomorphic encryption | C. Gentry | compute on encrypted data | SEAL, OpenFHE | link |
| 2020 | Indistinguishability obfuscation | A. Jain, H. Lin, A. Sahai | iO from well-founded assumptions | — | link |

In the net: A key exchange and a zero-knowledge proof are meetings (x,−x, carry 0): the two passages close against each other and nothing passes as carry to the observer. A one-way function is a window that closes in one direction of the strand and stays open in the other (N3); in proof-of-work consensus the longest strand holds, and its closed blocks are fixed by first passage. Corresponded.

**Semantics and logic of programs**

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1967/1969 | Floyd–Hoare logic | R. Floyd; C.A.R. Hoare | pre/postcondition triples | Dafny | link |
| 1969–1971 | Denotational semantics | D. Scott, C. Strachey | programs denote domain elements | — | link |
| 1969/1978/1982 | Hindley–Milner type inference | R. Hindley; R. Milner; L. Damas | principal polymorphic types | ML, Haskell | link |
| 1972/1974 | System F | J.-Y. Girard; J. Reynolds | polymorphic lambda calculus | GHC Core | link |
| 1972–1984 | Martin-Löf type theory | P. Martin-Löf | dependent types | Agda | link |
| 1972/1979 | LCF | R. Milner (Stanford LCF); Gordon, Milner, Wadsworth (Edinburgh LCF) | abstract theorem type, tactics | HOL, Isabelle | link |
| 1975 | Weakest preconditions | E. Dijkstra | predicate transformers | — | link |
| 1975–1977 | Algebraic specification | ADJ group; Guttag | ADTs as initial algebras | OBJ, Maude | link |
| 1976 | Dynamic logic | V. Pratt | modal logic with program modalities | KeY | link |
| 1977 | PCF and full abstraction | G. Plotkin; R. Milner | denotational vs operational agreement | — | link |
| 1977 | Abstract interpretation | P. Cousot, R. Cousot | sound approximation via Galois connections | Astrée | link |
| 1978/1990 | Optimal lambda reduction | J.-J. Lévy; J. Lamping | share redex families | BOHM | link |
| 1981 | Structural operational semantics | G. Plotkin | inductive transition rules | — | link |
| 1985/1988 | Calculus of Constructions | T. Coquand, G. Huet | higher-order dependent types | Coq/Rocq | link |
| 1987 | Natural semantics | G. Kahn | big-step evaluation rules | — | link |

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1987 | Linear logic | J.-Y. Girard | resource-sensitive logic | — | link |
| 1988/2000 | Coalgebra / coinduction | P. Aczel; J. Rutten | final coalgebras for infinite behaviour | — | link |
| 1989/1992 | Monads for effects | E. Moggi; P. Wadler | effects as monads | Haskell IO | link |
| 1989 | Geometry of interaction | J.-Y. Girard | normalization as dynamics of operators | — | link |
| 1990 | Interaction nets | Y. Lafont | local binary graph rewriting, strongly confluent | HVM | link |
| 1990 | Calculus of Inductive Constructions | T. Coquand, C. Paulin-Mohring | inductive types added | Coq, Lean | link |
| 1994 | Game semantics | Abramsky–Jagadeesan–Malacaria; Hyland–Ong | computation as dialogue | — | link |
| 2001–2009 | Algebraic effects and handlers | Plotkin, Power, Pretnar | effects as operations, handlers | OCaml 5, Koka | link |
| 2001–2002 | Separation logic | Reynolds, O'Hearn, Ishtiaq, Yang | local heap reasoning | Infer, Iris | link |
| 2005–2013 | Homotopy type theory | V. Voevodsky; Awodey–Warren | types as spaces; univalence | HoTT Coq library | link |
| 2015/2018 | Cubical type theory | Cohen, Coquand, Huber, Mörtberg | computational univalence | Cubical Agda | link |

In the net: A program text is a strand, its meaning is the closure that the strand reaches, and a verified proof is that closure made explicit (N2); a type is the context in which a term responds (N4). Pre/postconditions, linear resources and interaction nets are local closures in which each active pair closes once and passes no carry; a closed sub-proof passes up as one ring (x,x,x) to the next scale (N5). Corresponded.

## Concurrency and distributed computing

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1973 | Actor model | Hewitt, Bishop, Steiger | asynchronous message-passing actors | Erlang, Akka | link |
| 1974 | Kahn process networks | G. Kahn | deterministic processes on FIFO channels | Daedalus, AMD Versal AI Engines | link |
| 1974/1975 | Dataflow architecture | J. Dennis | instructions fire when operands arrive | MIT static dataflow prototype | link |
| 1977 | Temporal logic of programs | A. Pnueli | LTL for program behaviour | SPIN | link |
| 1978 | CSP | C.A.R. Hoare | synchronize on events | occam, Go | link |
| 1978 | Logical clocks | L. Lamport | happened-before ordering | Lamport timestamps in distributed databases | link |
| 1980 | CCS | R. Milner | algebraic process calculus | Concurrency Workbench | link |
| 1980/1982 | Byzantine generals | Lamport, Shostak, Pease | agreement needs n > 3f | PBFT | link |
| 1981 | Bisimulation | D. Park; R. Milner | behavioural equivalence | mCRL2 | link |
| 1981–1982 | Model checking | Clarke, Emerson; Queille, Sifakis | automatic check of temporal formulas | SPIN, NuSMV | link |
| 1982–1984 | ACP | J. Bergstra, J.W. Klop | axiomatic process algebra | mCRL2 | link |
| 1985 | FLP impossibility | Fischer, Lynch, Paterson | no deterministic async consensus with one crash | — | link |
| 1986/1990/1993 | Symbolic model checking (BDDs) | R. Bryant; Burch, Clarke, McMillan, Dill, Hwang; K. McMillan | decision diagrams for state sets | SMV | link |
| 1987/1990 | Linearizability | M. Herlihy, J. Wing | operations appear atomic | Jepsen | link |

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1987 | I/O automata | N. Lynch, M. Tuttle | composable automata | IOA toolkit | link |
| 1989/1992 | Pi-calculus | Milner, Parrow, Walker | mobile processes | Pict | link |
| 1989/1998 | Paxos | L. Lamport | majority-quorum consensus | Chubby, Spanner | link |
| 1990/1994 | Timed automata | R. Alur, D. Dill | automata with clocks | UPPAAL | link |
| 1991 | Wait-free hierarchy | M. Herlihy | consensus numbers | lock-free libraries | link |
| 1993 | Session types | K. Honda | types for protocols | Scribble | link |
| 1993 | Topological distributed computing | Herlihy–Shavit; Saks–Zaharoglou; Borowsky–Gafni | solvability via topology | — | link |
| 1994 | TLA | L. Lamport | temporal logic of actions | TLA+ | link |
| 1998 | Ambient calculus | L. Cardelli, A. Gordon | movement of nested boundaries | — | link |
| 2000/2002 | CAP theorem | E. Brewer; S. Gilbert, N. Lynch | consistency vs availability under partition | Cassandra | link |
| 2011 | CRDTs | Shapiro, Preguiça, Baquero, Zawirski | commutative merge of replicas | Automerge | link |
| 2014 | Raft | D. Ongaro, J. Ousterhout | leader-based replicated log | etcd | link |

In the net: Synchronisation, handshake and commit/abort agreement are meetings (x,−x, carry 0) between two strands; a crash or partition leaves a passage open, which is the content of FLP and CAP. In a logical clock order follows from first passage (N3), a quorum is the ring (x,x,x) in which the majority closes and passes up as one decision, and a temporal formula holds when every open passage eventually closes. Corresponded.

## Parallel models

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1978 | PRAM | S. Fortune, J. Wyllie | synchronous shared-memory processors | XMT | link |
| 1978 | Systolic arrays | H.T. Kung, C. Leiserson | rhythmic data pumping through grids | Google TPU | link |
| 1979 | NC | N. Pippenger; S. Cook | polylog depth parallel class | — | link |
| 1987/1992/2000 | LOCAL / CONGEST models | N. Linial; D. Peleg | neighbour-communication rounds | — | link |
| 1988 | External memory model | A. Aggarwal, J. Vitter | cost in block transfers | STXXL | link |
| 1990 | BSP | L. Valiant | supersteps with barrier | Pregel | link |
| 1993 | LogP | Culler, Karp, Patterson et al. | latency, overhead, gap, processors | — | link |
| 1994–1999 | Work stealing / Cilk | R. Blumofe, C. Leiserson | work-span scheduling | Cilk, TBB | link |
| 1999 | Cache-oblivious model | Frigo, Leiserson, Prokop, Ramachandran | optimal across unknown hierarchies | FFTW | link |
| 2004/2010/2013 | MapReduce / MPC model | Dean, Ghemawat; Karloff–Suri–Vassilvitskii (MRC); Beame–Koutris–Suciu (MPC) | map and reduce rounds | Hadoop, Spark | link |
| 2004 | Population protocols | D. Angluin et al. | anonymous agents interacting pairwise | DNA strand displacement | link |

In the net: In a parallel step many trit windows close side by side; a barrier or superstep is the point at which they close together and their carry passes to the next scale (N5). Depth counts scales, work counts windows, and by Theorem C1 at most two passages per scale stay open, which suggests depth of order $\log_3 n$ for problems whose windows close locally; this is a hypothesis, not a result. Corresponded.

## Data, logic programming and rewriting

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1968 | L-systems | A. Lindenmayer | parallel string rewriting for growth | L-Studio | link |
| 1970 | Relational model | E.F. Codd | data as relations | SQL databases | link |
| 1970 | Knuth–Bendix completion | D. Knuth, P. Bendix | equations to confluent rewrite systems | Waldmeister | link |
| 1972–1974 | Logic programming | R. Kowalski; A. Colmerauer, P. Roussel | logic as programming language | Prolog, WAM | link |
| 1973 | Algebraic graph rewriting | Ehrig, Pfender, Schneider | double-pushout rewriting | AGG | link |
| 1977–1985 | Datalog | Gallaire, Minker; Chandra–Harel | recursive Horn queries | Soufflé | link |
| 1977–1992 | Term rewriting theory | Huet; Dershowitz; Klop | confluence and termination | Maude | link |
| 1987 | Constraint logic programming | J. Jaffar, J.-L. Lassez | unification generalized to constraints | ECLiPSe | link |
| 1988 | Answer set programming | M. Gelfond, V. Lifschitz | stable models | clingo | link |
| 1992 | Rewriting logic | J. Meseguer | concurrency as deduction | Maude | link |
| 1996/2001 | CDCL SAT solving | Marques-Silva & Sakallah; Chaff | clause learning | MiniSat, Z3 | link |

In the net: A rewrite step closes one window of a term, and the result passes up; confluence means every order of closure reaches the same fixed point (N2). A query or logic program ends in closure when no further fact or clause opens a new window; in clause learning a failed closure remains as memory, and later passages read relative to it (N3). Corresponded.

## Alternative machine models, cellular and molecular

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1970/1980 | Storage modification machine | A. Schönhage | pointer machine | — | link |
| 1970/1982 | Game of Life universality | Conway; Berlekamp–Conway–Guy | Life is Turing-complete | Golly | link |
| 1973 | RAM / RASP model | S. Cook, R. Reckhow | random-access machine costs | — | link |
| 1973/1982 | Reversible computation | C. Bennett; Fredkin–Toffoli | logically reversible computing | adiabatic CMOS | link |
| 1985/2000 | Abstract state machines | Y. Gurevich | ASM thesis | AsmL | link |
| 1994 | DNA computing | L. Adleman | Hamiltonian path solved with DNA molecules | wet-lab DNA computers | link |
| 1998 | Tile self-assembly (aTAM) | E. Winfree | Wang tiles self-assemble as universal computation | DNA tile crystals | link |
| 1998 | Membrane (P) systems | G. Păun | nested membranes with multiset rewriting | simulators | link |
| 2004 | Rule 110 universality | M. Cook | 1D CA universal | — | link |
| 2006/1997 | Spiking neural P systems / neuromorphic models | Ionescu, Păun, Yokomori; Maass | spike timing as computational resource | Loihi, SpiNNaker | link |
| 2008 | Chemical reaction networks as computers | Soloveichik, Cook, Winfree, Bruck | Turing-universal computation with molecular counts and arbitrarily small error | DNA strand displacement (Soloveichik–Seelig–Winfree 2010) | link |

In the net: A cellular automaton, a tile assembly and a chemical reaction network are strands laid out in space, each cell window closing locally and its carry passing to the neighbours (N5). Reversible computation is the case in which no window is erased: every closure can be read back along the strand, so memory of first passage is kept in full (N3). Corresponded.

## Interaction and hypercomputation

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1992/2002 | Malament–Hogarth spacetime computing | Hogarth; Etesi–Németi | relativistic spacetimes allow infinitely many steps in finite observer time | — | link |
| 1994–1995 | Analog recurrent neural networks | H. Siegelmann, E. Sontag | real-weighted networks compute beyond Turing (P/poly) | — | link |
| 1997 | Interactive computation | P. Wegner | interaction more expressive than algorithms | — | link |
| 1999 | Hypercomputation (term) | J. Copeland, D. Proudfoot | models computing non-Turing-computable functions | — | link |
| 2000/2004 | Persistent Turing machines | D. Goldin; Goldin, Smolka, Attie, Sonderegger | multitape machine keeps memory between interaction rounds | — | link |
| 2000 | Infinite time Turing machines | J. Hamkins, A. Lewis | transfinite ordinal steps of computation | — | link |

In the net: In interaction a passage stays open between the machine and its context, so the response is a state function of the history and not of the input alone (N4). Hypercomputation models assume closure after infinitely many windows; in the net the strand is finite at every scale, and what does not close passes as carry to the next scale without end (N5). Corresponded.

## Real computation

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1981/1989 | Pour-El–Richards wave equation result | M. Pour-El, I. Richards | computable initial data with noncomputable evolution | — | link |
| 1985/2000 | Computable analysis / Type-2 theory of effectivity | K. Weihrauch; Kreitz–Weihrauch | reals via infinite representations on Type-2 machines | iRRAM | link |
| 1986–1997 | Exact real arithmetic | Boehm; Edalat; Escardó | arbitrary-precision reals via lazy streams/domains | RealLib, iRRAM | link |
| 1989 | BSS model | L. Blum, M. Shub, S. Smale | machines computing over the reals or arbitrary rings | — | link |
| 1992/2011 | Weihrauch reducibility | K. Weihrauch; Brattka–Gherardi | classify uncomputability of theorems | — | link |

In the net: A real number is a strand of trits that never closes at a finite scale; a computable real is one whose open passages carry in a controlled way, so each scale fixes one more digit (N5). Equality of two reals stays open, which is the source of the uncomputability results for real machines and wave equations. Corresponded.

## Algorithms and analysis models

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1974–1976 | Approximation algorithms / PTAS | D. Johnson; Sahni; Christofides | provable approximation ratios for NP-hard problems | — | link |
| 1979/1984 | Ellipsoid / interior-point LP | L. Khachiyan; N. Karmarkar | linear programming in polynomial time | CPLEX, Gurobi | link |
| 1985 | Amortized analysis / splay trees | R. Tarjan; D. Sleator | average cost over operation sequences | — | link |
| 1985 | Online algorithms / competitive analysis | D. Sleator, R. Tarjan | online decisions compared with offline optimum | LRU caching analysis | link |
| 1985/2007 | Flajolet–Martin / HyperLogLog | P. Flajolet, G. Martin; Flajolet et al. | count distinct elements with tiny memory | Redis, BigQuery | link |
| 1995 | Semidefinite relaxation (Goemans–Williamson) | M. Goemans, D. Williamson | SDP plus random hyperplane: 0.878-approximation for MAX-CUT | — | link |
| 1996 | Streaming algorithms | N. Alon, Y. Matias, M. Szegedy | frequency moments in sublinear space | network monitors | link |
| 1996/1998 | Property testing | Rubinfeld–Sudan; Goldreich–Goldwasser–Ron | closeness to a property from few queries | — | link |
| 2001/2004 | Smoothed analysis | D. Spielman, S. Teng | worst case under small random perturbations; explains simplex | — | link |
| 2004 | Spectral graph sparsification / Laplacian solvers | D. Spielman, S. Teng | nearly-linear-time solutions of Laplacian systems | — | link |
| 2005/1984 | Sketching (Count-Min, Johnson–Lindenstrauss) | G. Cormode, S. Muthukrishnan; Johnson–Lindenstrauss | compact linear summaries for approximate queries | stream processors | link |

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 2018 | Algorithms with predictions | Lykouris–Vassilvitskii; Purohit et al. | online algorithms using ML advice robustly | — | link |
| 2022 | Almost-linear max flow | Chen, Kyng, Liu, Peng, Probst Gutenberg, Sachdeva | maximum flow in m^(1+o(1)) time | — | link |
| 2025 | Shortest paths beyond Dijkstra's sorting barrier | Duan, Mao, Mao, Shu, Yin | directed SSSP faster than O(m + n log n) | — | link |

In the net: An algorithm bound counts windows to closure; amortized and competitive analysis count carry across a sequence of passages rather than per window (N5). A sketch or streaming summary holds only the open passages per scale, which under Theorem C1 costs memory of order $\log_3 n$ per counter; actual sketches differ (HyperLogLog registers need about log log n bits, Count-Min size is set by its error bounds). Corresponded.

### Algorithmic game theory

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1988/1994/2011 | Total search problem classes (PLS, PPA, PPP, CLS) | Johnson–Papadimitriou–Yannakakis; Papadimitriou; Daskalakis–Papadimitriou | total NP search problems classified by existence argument | — | link |
| 1994/2006 | PPAD and complexity of Nash | C. Papadimitriou; Daskalakis–Goldberg–Papadimitriou; Chen–Deng | computing Nash equilibria is PPAD-complete | — | link |
| 1999 | Price of anarchy | E. Koutsoupias, C. Papadimitriou | equilibrium cost versus social optimum | — | link |
| 1999 | Algorithmic mechanism design | N. Nisan, A. Ronen | incentive-compatible mechanisms that are computationally efficient | ad auctions | link |

In the net: An equilibrium is a fixed point at which each player's window closes against the others (N2); a total search problem guarantees that such a closure exists without a short strand to it. The price of anarchy measures the carry that remains open between the equilibrium closure and the social optimum. Corresponded.

## Learning theory

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1964/2000 | Solomonoff induction / AIXI | R. Solomonoff; M. Hutter | Bayesian prediction over all programs; optimal universal agent | MC-AIXI approximation | link |
| 1967 | Gold identification in the limit | E.M. Gold | learning languages from examples in the limit | — | link |
| 1968/1971 | VC dimension | V. Vapnik, A. Chervonenkis | capacity measure governing uniform convergence | SVMs | link |
| 1984 | PAC learning | L. Valiant | efficient probably-approximately-correct learning from samples | — | link |
| 1988 | Mistake-bound / Littlestone dimension | N. Littlestone | online learning mistakes; Winnow algorithm | — | link |
| 1990/1995–1997 | Boosting | R. Schapire; Y. Freund, R. Schapire | weak learners combine into strong learners (AdaBoost) | later boosting libraries (XGBoost: gradient boosting) | link |
| 1993 | Statistical query model | M. Kearns | learning from noisy statistical queries | — | link |
| 1994/2003 | Multiplicative weights / online convex optimization | Littlestone–Warmuth; Zinkevich | regret minimization by weight updates | — | link |
| 1995/2019/2023 | Expressivity of neural networks and transformers | Siegelmann–Sontag; Pérez–Barceló–Marinković; Merrill–Sabharwal | recurrent/transformer networks as computation models; constant-depth transformers in TC0 | LLMs | link |
| 2019/2022 | Private and online learnability equivalence | Alon, Bun, Livni, Malliaris, Moran | private PAC learnable iff finite Littlestone dimension | — | link |

In the net: Learning is the convergence of a strand of hypotheses to closure on the data; identification in the limit and mistake bounds count the passages before that closure is fixed by memory (N3). Capacity measures (VC, Littlestone) count the windows that can stay open, and in boosting weak closures pass up as one ring (x,x,x) with a stronger carry. Corresponded.

## Quantum computation

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1980/1982 | Quantum simulation proposal | Y. Manin; R. Feynman | quantum systems need quantum computers for efficient simulation | — | link |
| 1985 | Quantum Turing machine | D. Deutsch | universal quantum computer model | — | link |
| 1989/1993 | Quantum circuit model | D. Deutsch; A. Yao | quantum circuits efficiently simulate QTMs (Yao); equivalent under uniformity | IBM Q, Google Sycamore | link |
| 1993/1997 | BQP / Bernstein–Vazirani | E. Bernstein, U. Vazirani | quantum complexity theory; oracle separation from BPP | — | link |
| 1994 | Simon's algorithm | D. Simon | exponential oracle speedup for hidden XOR mask | — | link |
| 1994 | Shor's algorithm | P. Shor | polynomial-time quantum factoring and discrete log | small demos on quantum hardware | link |
| 1995–1997 | Quantum error correction / threshold theorem | Shor; Steane; Aharonov–Ben-Or; Kitaev | fault-tolerant quantum computation below an error threshold | Google Willow surface code (2024) | link |
| 1996 | Grover search | L. Grover | quadratic speedup for unstructured search | — | link |
| 1997/2003 | Topological quantum computation | A. Kitaev; Freedman–Kitaev–Larsen–Wang | braiding anyons gives fault-tolerant gates | Microsoft Majorana 1 (2025) | link |
| 1999/2013 | QMA and quantum PCP | A. Kitaev; Aharonov–Arad–Vidick | local Hamiltonian problem is QMA-complete | — | link |
| 2000/2004 | Adiabatic quantum computation | Farhi, Goldstone, Gutmann, Sipser; Aharonov et al. | slow Hamiltonian evolution to ground state; polynomially equivalent | D-Wave (annealing) | link |

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 2001 | Measurement-based (one-way) quantum computation | R. Raussendorf, H. Briegel | computation by single-qubit measurements on a cluster state | photonic platforms | link |
| 2001 | KLM linear optics quantum computing | E. Knill, R. Laflamme, G. Milburn | scalable QC with linear optics and measurement | PsiQuantum | link |
| 2003–2004 | Quantum walk algorithms / query complexity | Childs et al.; Ambainis | quantum walks give speedups (glued trees, element distinctness) | — | link |
| 2009 | HHL linear systems algorithm | A. Harrow, A. Hassidim, S. Lloyd | exponential speedup for sparse linear systems (with caveats) | — | link |
| 2011 | BosonSampling / quantum advantage | S. Aaronson, A. Arkhipov | sampling hardness as evidence for quantum advantage | Jiuzhang | link |
| 2018 | Classical verification of quantum computation | U. Mahadev | classical verifier checks BQP using LWE | — | link |
| 2018 | Raz–Tal oracle separation BQP vs PH | R. Raz, A. Tal | oracle relative to which BQP is not in PH | — | link |
| 2018 | Dequantization | E. Tang | classical sampling algorithms match quantum recommender speedups | — | link |
| 2020 | MIP* = RE | Ji, Natarajan, Vidick, Wright, Yuen | entangled-prover interactive proofs decide halting; refutes Connes embedding | — | link |
| 2023 | Regev quantum factoring | O. Regev | factoring with ~n^1.5 gates, multidimensional Shor variant | — | link |

In the net: In a qubit a trit window stays open in superposition until it closes at measurement (N2); interference is the meeting (x,−x, carry 0) of amplitudes that cancel. In error correction local closures pass up as rings (x,x,x) at each code scale, and below threshold the open passages per scale stay bounded, the form of Theorem C1. Corresponded.

## Formal languages and automata

| Year | Theory | Person(s) | Core idea | Implementation | Source |
|---|---|---|---|---|---|
| 1957/1958; 1964 | Myhill–Nerode minimization and Brzozowski derivatives | J. Myhill, A. Nerode; J. Brzozowski | regular expression derivatives give automata | regex engines | link |
| 1961/1963 (weighted theory 1970s+) | Weighted automata / probabilistic automata | M. Schützenberger; M. Rabin | automata with semiring weights | OpenFst | link |
| 1962/1969 | Büchi/Rabin automata on infinite objects | J.R. Büchi; M. Rabin | S2S decidable via tree automata | Spot library | link |
| 1965–1968 | Tree automata / tree transducers | Thatcher, Wright; Doner | automata over terms; MSO equivalence | XML schema validators | link |
| 1983–2004 | Graph minor theorem / algorithms | N. Robertson, P. Seymour | minor-closed families finitely characterized; cubic-time membership | — | link |
| 1983–1987 | Synchronous languages | Berry (Esterel); Caspi–Halbwachs (Lustre) | synchrony hypothesis: reactions take zero time | SCADE (Airbus) | link |
| 1990 | Courcelle's theorem | B. Courcelle | MSO properties linear-time on bounded-treewidth graphs | — | link |
| 1993/1996 | Hybrid automata | Alur, Courcoubetis, Henzinger, Ho; T. Henzinger | discrete control with continuous dynamics | SpaceEx | link |
| 1997 | Functional reactive programming | C. Elliott, P. Hudak | time-varying values and events as first-class | Fran, Yampa, reactive-banana | link |

In the net: An automaton is the smallest set of contexts in which a strand of symbols responds (N4); under minimization strands that reach the same closure count as one. Automata on infinite words and trees accept when the strand returns to closure infinitely often; in synchronous and hybrid models each reaction closes within one tick and the state passes as carry to the next (N5). Corresponded.

## Catalogue 4 — AI, learning machines, physical and unconventional computing

179 items accepted from 228 candidates; 21 corrected by the judge, 0 rejected, 49 left out as already in Catalogues 1–3 (respondent–judge protocol: every candidate checked against a source by an independent judge; all fields confirmed on opened pages after re-check).

## Symbolic, cognitive and architectural AI

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1951 | Deontic logic | von Wright | Modal logic of obligation and permission for normative reasoning | Rule engines, legal reasoning systems | link |
| 1963/1969 | Situation calculus | McCarthy, Hayes | First-order logic of actions, situations and fluents | Theorem provers; Golog (1997) | link |
| 1964 | DENDRAL | Feigenbaum, Lederberg, Buchanan, Djerassi | First knowledge-based system; heuristics infer molecular structure | Lisp (first program by Georgia Sutherland) | link |
| 1966–69 | Semantic networks | Quillian; Collins | Knowledge as labelled graph; spreading activation | TLC program | link |
| 1971–80 | Blackboard architecture | Erman, Lesser, Reddy et al. | Independent knowledge sources cooperate via shared blackboard | Hearsay-II speech system | link |
| 1971 | STRIPS planning | Fikes, Nilsson | Actions as precondition/ add/delete lists; state-space planning | Shakey the robot (SRI) | link |
| 1972 | Production systems | Newell, Simon (1972; Newell 1973), after Post 1943 | Condition-action rules fired from working memory | OPS5 (early 1980s); Rete (Forgy 1974/1979/1982) | link |
| 1972–76 | Expert systems (MYCIN) | Shortliffe, Buchanan, Cohen | Backward-chaining rules with certainty factors for diagnosis | Lisp on DEC KI10 PDP-10 | link |
| 1973–79 | Lisp machines | Greenblatt, Knight (MIT AI Lab) | Hardware tagged architecture dedicated to symbolic processing | CADR, Symbolics 3600 | link |
| 1974 | Frames | Minsky | Stereotyped structured slots with defaults for situations | FRL, KRL, KL-ONE | link |
| 1976 | Physical symbol system hypothesis | Newell, Simon | Symbol manipulation is necessary and sufficient for general intelligence | Theory; LISP/ IPL systems | link |

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1977 | Scripts | Schank, Abelson | Stereotyped event sequences guide understanding | SAM program (Yale) | link |
| 1980 | Non-monotonic / default logic | Reiter; McCarthy (circumscription) | Conclusions withdrawn when new information arrives | Answer-set solvers | link |
| 1982–93 | Case-based reasoning | Schank; Kolodner | Solve new problems by retrieving and adapting past cases | CYRUS, CHEF | link |
| 1983–87 | SOAR | Laird, Newell, Rosenbloom | Unified cognition: problem spaces, impasses, chunking | C/C++ software | link |
| 1984 | Cyc | Lenat | Hand-encoded common-sense ontology with inference engine | CycL software | link |
| 1986 | Subsumption architecture | Brooks | Layered reactive behaviours without central world model | Allen, Genghis robots | link |
| 1986 | Society of Mind | Minsky | Intelligence from many simple interacting agents | Theory | link |
| 1986 | Decision-tree learning (ID3/ C4.5) | Quinlan | Recursive splits by information gain | Software | link |
| 1987/1995 | BDI agents | Bratman; Rao, Georgeff | Agents reason via beliefs, desires, intentions | PRS, JACK, Jason | link |
| 1990 | Inductive logic programming | Muggleton | Learn logic programs from examples plus background knowledge | Progol (Muggleton 1995), Aleph (Srinivasan 2001) | link |
| 1993 (ACT 1976) | ACT-R | Anderson | Declarative chunks plus production rules with subsymbolic activation | Lisp/Python software | link |

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 2006 | LIDA | Franklin | Global-workspace cognitive cycle implemented as architecture | Java framework | link |
| 2008 | OpenCog / AtomSpace | Goertzel | Hypergraph knowledge store with probabilistic logic networks | Software | link |
| 2009/2019 | Neuro-symbolic AI | Garcez, Lamb; Mao et al. (NS-CL) | Neural perception combined with symbolic reasoning | Software (e.g. AlphaGeometry 2024) | link |
| 1997 | CLARION | Sun (Ron) | Dual explicit/implicit (symbolic/subsymbolic) levels | Software | link |

In the net: a rule system is a strand, one ordered path of rule firings; a goal reached or a proof found is a closure, and a subgoal left open is carry to the next level of the problem space (N5). Chunking in SOAR and case retrieval are memory in the N3 sense: the first solved passage fixes the direction for later passages. Corresponded.

## Connectionism and deep learning, including test-time compute and reasoning effort

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1961 | Learning matrix | Steinbuch | Associative matrix memory | Lernmatrix hardware | link |
| 1965 | Group Method of Data Handling | Ivakhnenko, Lapa | First deep multilayer trained networks by layer-wise selection | Software | link |
| 1969 | Perceptrons critique | Minsky, Papert | Single-layer limits (XOR, connectedness) | Theory | link |
| 1970/1974/1986 | Backpropagation | Linnainmaa; Werbos; Rumelhart, Hinton, Williams | Reverse-mode chain rule trains multilayer networks | Software; GPUs | link |
| 1975 | CMAC | Albus | Cerebellum-inspired tile-coded associative memory | Robot control software/ hardware | link |
| 1976/1987 | Adaptive resonance theory | Grossberg, Carpenter | Top-down expectation match stabilises learning (stability-plasticity) | ART1/2 software and analog chips | link |
| 1980 | Neocognitron | Fukushima | Hierarchical S/C cell layers for shift-invariant recognition | Software | link |
| 1982 | Hopfield network | Hopfield | Energy-minimising recurrent net as associative memory | Analog circuits; optical implementations | link |
| 1982 | Self-organizing map | Kohonen | Competitive learning forms topology-preserving maps | Software | link |
| 1985 | Boltzmann machine | Hinton, Sejnowski (Ackley) | Stochastic energy-based net learns distributions | Software; p-bit/annealer hardware | link |
| 1986/2002 | Restricted Boltzmann machine | Smolensky; Hinton | Bipartite Boltzmann trained by contrastive divergence | Software; D-Wave sampling | link |
| 1986–90 | Recurrent nets (Elman/Jordan, BPTT) | Jordan; Elman; Werbos | Hidden state carries temporal context | Software | link |

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1988 | Sparse distributed memory | Kanerva | High-dimensional binary addresses as content-addressable memory | Software/ hardware prototypes | link |
| 1989 | Convolutional neural network (LeNet) | LeCun et al. | Weight-shared convolutions trained with backprop | AT&T DSP boards; GPUs (AlexNet 2012) | link |
| 1991/2017 | Mixture of experts | Jacobs, Jordan, Nowlan, Hinton; Shazeer et al. | Gating network routes inputs to specialist sub-networks | GPU/TPU clusters | link |
| 1994–2009 | Hyperdimensional / vector-symbolic computing | Kanerva; Plate (HRR); Gayler | Bind/bundle very high-dimensional vectors as symbols | Software; in-memory HDC chips | link |
| 1997 | LSTM | Hochreiter, Schmidhuber | Gated memory cell avoids vanishing gradients | Software; GPUs | link |
| 2001 | Echo state network | Jaeger | Fixed random recurrent reservoir; train only readout | Software; physical reservoirs | link |
| 2002 | Liquid state machine | Maass, Natschläger, Markram | Spiking recurrent "liquid" with trained readout | Software; neuromorphic chips | link |
| 2004/2007 | Hierarchical temporal memory | Hawkins, George | Cortical-column sequence memory with sparse distributed representations | NuPIC software | link |
| 2006 | Deep belief networks | Hinton, Osindero, Teh | Greedy layer-wise stacked RBM pretraining | Software | link |
| 2006/2013 | Autoencoders / VAE | Hinton, Salakhutdinov; Kingma, Welling | Compress-reconstruct; variational latent generative model | Software | link |
| 2006/2022 | Energy-based models / JEPA | LeCun et al. | Learn energy over compatible pairs; predict in latent space | Software (I-JEPA, V-JEPA) | link |

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 2009 | Graph neural networks | Scarselli, Gori et al. | Message passing over graph structure | Software | link |
| 2013 | Word embeddings (word2vec) | Mikolov et al. | Words as vectors from context prediction | Software | link |
| 2014 | Neural Turing machine / differentiable memory | Graves, Wayne, Danihelka | Neural controller with differentiable external memory | Software | link |
| 2014 | Attention mechanism | Bahdanau, Cho, Bengio | Learned soft alignment over input positions | Software | link |
| 2014 | Generative adversarial networks | Goodfellow et al. | Generator versus discriminator minimax game | GPUs | link |
| 2015 | Residual networks | He et al. | Skip connections enable very deep training | GPUs | link |
| 2015/2020 | Diffusion models | Sohl-Dickstein et al.; Ho, Jain, Abbeel | Learn to reverse gradual noising process | GPUs | link |
| 2015/2022 | Normalizing flows / flow matching | Rezende, Mohamed; Lipman et al. | Invertible or learned-velocity transport between distributions | Software | link |
| 2016/2020 | Modern Hopfield networks | Krotov, Hopfield; Ramsauer et al. | Higher-order interactions raise capacity (2016); exponential capacity (Demircigil 2017, Ramsauer 2020); update = attention | Software | link |
| 2017 | Capsule networks | Sabour, Frosst, Hinton | Vector capsules with routing-by-agreement | Software | link |
| 2017 | Transformer | Vaswani et al. | Self-attention replaces recurrence; parallel sequence | GPUs/TPUs | link |

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 2017/2022 | RLHF | Christiano et al.; Ouyang et al. | Reward model from human preferences tunes policy | Software | link |
| 2018/2020 | Large language models / GPT pretraining | Radford et al. (OpenAI); Brown et al. | Next-token prediction at scale yields in-context learning | GPU/TPU clusters | link |
| 2018 | Neural ODEs | Chen, Rubanova, Bettencourt, Duvenaud | Continuous-depth network as learned differential equation | Software | link |
| 2019/2021 | Score-based generative models (SDE) | Song, Ermon et al. | Learn gradient of log-density; sample by Langevin/SDE | GPUs | link |
| 2020/2022 | Neural scaling laws | Kaplan et al.; Hoffmann et al. (Chinchilla) | Loss falls as power law in parameters, data, compute | Empirical on GPU clusters | link |
| 2020 | Liquid time-constant networks | Hasani, Lechner et al. | Neuron time constants vary with input (C. elegans-inspired) | Software | link |
| 2020 | Neural cellular automata | Mordvintsev et al. | Learned local update rules grow and regenerate patterns | GPUs | link |
| 2021/2023 | State-space models (S4, Mamba) | Gu, Goel, Ré; Gu, Dao | Linear-time structured/selective state recurrence for sequences | GPUs (hardware-aware scan) | link |
| 2022 | Chain-of-thought prompting | Wei et al. | Intermediate reasoning tokens improve multi-step answers | LLMs | link |
| 2024 | Kolmogorov–Arnold networks | Liu, Tegmark et al. | Learnable spline activations on edges instead of nodes | Software | link |

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 2024 | Test-time compute scaling | OpenAI o1; Snell et al. | More inference-time reasoning computation substitutes for parameters (on problems of moderate difficulty) | LLMs on GPU clusters | link |
| 2025 | Reasoning effort vs problem complexity | Estermann, Wattenhofer et al. | Reasoning tokens scale with complexity then collapse | LLMs (Tents puzzle) | link |
| 2026 | Reasoning-effort economics | Kaiser Sun, Kwon, Hasan, Zhu, Zhou, Bianchi | Effort a "credence good"; start low, escalate after verified failure | Proprietary/ open-weight LLM agents | link |
| 2026 | Deep-thinking tokens | W.-L. Chen, Peng, Tan, Zhao, B. Chen, Lin, Go, Meng | Measure reasoning effort by depth, not token length | LLMs | link |

In the net: a Hopfield or Boltzmann network settles into an energy minimum, which is closure; trained weights are memory as shape (N3), and a residual or gated path is carry passed to the next layer (N5). In attention and chain-of-thought the whole history is read again at each step, which matches the Sun et al. 2026 finding that agent input cost grows as n^1.8–1.9 in calls; Theorem C1 keeps at most 2 open passages per scale, and hypothesis H1 predicts that an agent context built this way lowers the read cost towards $\log_3 n$ per step; BitNet b1.58 weights are the balanced-ternary set $\{-1, 0, +1\}$. Corresponded.

## Statistical learning theory

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1992/1995 | Support vector machines / kernel methods | Vapnik, Cortes; Boser, Guyon | Maximum-margin separation in kernel feature space | Software | link |
| 2003 | Gödel machine | Schmidhuber | Self-referential agent rewrites itself after proving improvement | Theory | link |

In the net: a maximum-margin separator is a meeting closure, two complementary classes (x, −x) with the margin as the standing 0 between them. A self-rewriting Gödel machine is a strand whose accepted rewrites fix direction as memory (N3). Corresponded.

## Probabilistic, causal and decision-theoretic

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1966 | Hidden Markov models | Baum, Petrie | Latent Markov states emit observations; Baum–Welch training | Speech recognisers (IBM, Dragon) | link |
| 1985/1988 | Bayesian networks | Pearl | DAG factorises joint distribution; belief propagation | Software | link |
| 1988/1998 | Reinforcement learning / TD learning | Sutton, Barto | Learn from reward via temporal-difference prediction | Software | link |
| 1989 | Q-learning | Watkins | Off-policy learning of state-action values | Software | link |
| 1992 | TD-Gammon | Tesauro | Self-play TD(λ) neural net reaches master backgammon | IBM Research (T.J. Watson) | link |
| 1995/2000 | Causal calculus (do-calculus) | Pearl | Interventions and counterfactuals from causal graphs | Software | link |
| 2006 | Monte Carlo tree search | Coulom; Kocsis, Szepesvári (UCT) | Sampled rollouts with bandit-based tree expansion | Software | link |
| 2013/2015 | Deep Q-network | Mnih et al. (DeepMind) | Deep CNN Q-learning from pixels on Atari | GPUs | link |
| 2016/2017/2019 | AlphaGo / AlphaZero / MuZero | Silver et al. (DeepMind) | Policy/value nets plus Monte Carlo tree search; self-play | TPUs | link |

In the net: a hidden state, a value function and a belief are context, a response that is a state function (N4). Value iteration and temporal-difference learning end in a fixed point, which is closure; the unresolved error passes as carry to the next step or the next level of a search tree (N5). Corresponded.

## Evolutionary, fuzzy, swarm and immune

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1964–73 | Evolution strategies | Rechenberg, Schwefel | Mutation with self-adaptive step sizes on real vectors | First on physical nozzle/plate experiments | link |
| 1975 | Genetic algorithms | Holland | Selection, crossover, mutation on bit-strings (schema theorem) | Software | link |
| 1986/2002 | Artificial immune systems | Farmer, Packard, Perelson; de Castro | Clonal selection and negative selection for learning | Software | link |
| 1987 | Boids / flocking | Reynolds | Three local rules generate collective flocking | Software | link |
| 1989/2014 | Swarm robotics | Beni, Wang; Rubenstein, Cornejo, Nagpal (Kilobots) | Many simple robots self-organise collectively | Kilobot 1024-robot swarm | link |
| 1992 | Genetic programming | Koza | Evolve program trees | Software; Beowulf clusters | link |
| 1992 | Ant colony optimisation | Dorigo | Pheromone-trail reinforcement finds short paths | Software | link |
| 1995 | Particle swarm optimisation | Kennedy, Eberhart | Particles move toward personal and global bests | Software | link |
| 1996/2002 | Evolvable hardware / evolution in materio | Thompson; Miller | Evolve configurations of physical chips or materials | Xilinx XC6216 (XC6200); carbon-nanotube composites later (Durham, 2010s) | link |
| 2002 | NEAT / neuroevolution | Stanley, Miikkulainen | Evolve network weights and topology together | Software | link |

In the net: under selection what closes stays and the remainder passes as carry to the next generation (N5); a stabilised genotype or a swarm consensus is closure. Pheromone trails and immune memory are first passages that fix direction (N3); fuzzy sets and evolutionary programming are in earlier catalogues. Corresponded.

### Brain and consciousness models

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1971/1991 | Holonomic brain theory | Pribram (Bohm) | Memory as distributed interference patterns | Theory; holographic optics analogy | link |
| 1987 | Neural Darwinism | Edelman | Selection among neuronal groups with re-entry | Darwin robot series | link |
| 1988/1998 | Global workspace theory | Baars; Dehaene (neuronal GW) | Consciousness as broadcast across specialist modules | LIDA; GW deep-learning models | link |
| 1994/1996 | Orch-OR | Penrose, Hameroff | Non-computable quantum gravity collapse in microtubules | Theory | link |
| 1999 | Predictive coding | Rao, Ballard (Srinivasan 1982) | Cortex sends predictions down, errors up | Software models | link |
| 2004 | Bayesian brain | Knill, Pouget; Helmholtz | Perception as probabilistic inference | Theory | link |
| 2004 | Integrated information theory | Tononi | Consciousness equals integrated information Φ | Theory; PyPhi software | link |
| 2006/2010 | Free-energy principle / active inference | Friston | Systems minimise variational free energy via perception and action | Software (SPM, pymdp) | link |
| 2019/2021 | Thousand Brains theory | Hawkins | Each cortical column learns full object models in reference frames | Monty software (Thousand Brains Project) | link |

In the net: prediction and prediction error form a meeting closure (x, −x, carry 0); an unexplained error passes upward as carry (N5), which is the structure of predictive coding and free-energy minimisation. Global broadcast and integrated information describe closure across many local windows at one scale, with re-entry as memory (N3). Corresponded.

## Analog, mechanical and fluidic

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1936 | Water integrator | Lukyanov | Water levels and flows solve differential equations | Hydraulic analog | link |
| 1947 | Electronic op-amp analog computers | Ragazzini, Randall, Russell; Philbrick | Op-amps integrate and sum voltages in real time | Philbrick, EAI, Telefunken machines | link |
| 1949 | MONIAC | Phillips | Hydraulic model computes the national economy | Water-flow machine | link |
| 1957/1959 | Fluidic logic | Horton; Bowles, Warren (Harry Diamond Labs) | Fluid jet interactions as logic gates | Fluidic amplifiers; microfluidic logic | link |
| 1987 | Field computation | MacLennan | Computation on continuous spatial fields, not discrete symbols | Optical/analog field processors (theory) | link |
| 1988 | Cellular neural networks | Chua, Yang | Locally coupled analog cells process images | CNN Universal Machine chips | link |

In the net: an analog machine is a strand of continuous state, and an integrator holds history as shape (N3). A flow network such as MONIAC reaches equilibrium when inflows and outflows cancel, a meeting closure; a cellular neural network settles to a fixed point per cell. Corresponded.

## Reversible and energy-limited

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1982 | Billiard-ball computer | Fredkin, Toffoli | Elastic ball collisions perform reversible logic | Thought model; soldier-crab experiment (Gunji 2011) | link |
| 1993 | Adiabatic / reversible CMOS | Younis, Knight; Frank | Slow charge recovery approaches zero dissipation | Pendulum, SCRL chips; Vaire Computing (2025) | link |

In the net: a billiard-ball collision and adiabatic charge recovery are meeting closures in which nothing is erased, so the carry is 0 and no energy is lost to erasure. Dissipation arises only where an open passage is discarded rather than passed on as carry (N5). Corresponded.

## Quantum

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1988/1995 | Quantum cellular automata | Grössing, Zeilinger; Watrous | Unitary local update rules on lattice | Theory; neutral-atom arrays | link |
| 1992 | Deutsch–Jozsa algorithm | Deutsch, Jozsa | Exponential oracle separation, constant vs balanced | NMR demonstration 1998 | link |
| 1998 | Quantum annealing | Kadowaki, Nishimori | Quantum tunnelling finds Ising ground states | D-Wave annealers (2011) | link |
| 1999/2001 | Continuous-variable quantum computing | Lloyd, Braunstein; Gottesman–Kitaev–Preskill | Compute with quadratures of bosonic modes | Xanadu squeezed-light chips | link |
| 2014 | Variational quantum algorithms (VQE/ QAOA) | Peruzzo et al.; Farhi, Goldstone, Gutmann | Hybrid classical optimiser tunes parameterised quantum circuit | NISQ devices | link |
| 2019 | Quantum supremacy / random circuit sampling | Arute et al. (Google) | Programmable chip samples beyond classical reach | Sycamore 53 qubits | link |

In the net: amplitude cancellation in interference is a meeting closure (x, −x); in Deutsch–Jozsa a balanced function cancels to 0 and a constant one does not. Annealing and the variational loop settle into an energy minimum, which is closure, and a quantum cellular automaton is a ring of identical local updates. Corresponded.

## Optical and photonic

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1960 | Optical / Fourier-optics computing | Cutrona, Leith (SAR processors) | Lenses perform Fourier transforms and correlation at light speed | Coherent optical processors | link |
| 1985 | Optical neural networks | Psaltis, Farhat | Holographic/optical implementation of neural associative memory | Optical Hopfield nets | link |
| 2017 | Integrated photonic deep learning | Shen, Harris et al. (MIT) | Mach–Zehnder meshes perform matrix multiplication | MIT programmable nanophotonic processor (silicon MZI mesh) | link |
| 2018 | Diffractive deep neural networks | Lin, Ozcan et al. | 3D-printed diffractive layers classify by light propagation | Passive THz diffractive plates | link |
| 2025 | Analog optical computer (AOC) | Kalinin et al. (Microsoft Research) | Fixed-point iteration in optics+analog electronics for AI and optimisation | Micro-LED/SLM/ photodetector prototype | link |

In the net: optical interference is a meeting closure in which fields cancel or add; a diffractive stack is a strand of fixed layers whose shape is memory (N3). The analog optical computer iterates to a fixed point, which is closure. Corresponded.

**Stochastic, thermodynamic and Ising**

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1954 | Parametron | Goto (Eiichi) | Phase of parametric oscillation stores binary state | PC-1 parametron computer | link |
| 1967 | Stochastic computing | Gaines; Poppelbaum group (after Ribeiro) | Numbers as probabilities of random bitstreams; AND multiplies | STL stochastic computer; RASCEL (Illinois) | link |
| 2016 | Coherent Ising machine | Inagaki, Marandi, McMahon, Takesue et al. (NTT) | Degenerate optical parametric oscillators encode coupled spins | 2000-spin fibre-loop CIM | link |
| 2017 | Probabilistic bits (p-bits) | Camsari, Datta et al. | Fluctuating bits enable invertible logic and sampling | Stochastic magnetic tunnel junctions (Borders 2019) | link |
| 2017 | Oscillator-based Ising machine | Wang, Roychowdhury | Sub-harmonic injection locking binarises coupled oscillator phases | CMOS LC oscillators | link |
| 2018–19 | Simulated bifurcation / digital annealer | Goto (Toshiba); Fujitsu | Classical bifurcation dynamics or digital annealing for optimisation | FPGA/GPU, Digital Annealer ASIC | link |
| 2019 | Thermodynamic computing (framework) | Conte, Hylton et al. (CCC) | Computation exploiting thermal fluctuations and dissipation | Theory/ roadmap | link |
| 2022 | Ising machines (survey) | Mohseni, McMahon, Byrnes | Physical systems minimise Ising energy for optimisation | Optical, electronic, spintronic platforms | link |
| 2023 | Thermodynamic AI / stochastic processing unit | Coles, Melanson, Aifer et al. (Normal Computing) | Langevin dynamics of RLC circuits sample and invert matrices | 8-cell SPU PCB | link |

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 2025 | Thermodynamic sampling unit / denoising thermodynamic models | Extropic (Jelinčič, Lockwood, Garlapati, Verdon, McCourt) | All-transistor probabilistic circuits run diffusion-like Gibbs sampling | X0 chip, XTR-0 platform (built); Z1 planned | link |

In the net: an Ising machine settles into a ground state, closure as an energy minimum; spin pairs of opposite sign are meeting closures. The two oscillation phases of a parametron plus the undriven state as standing 0 form the balanced set {−1, 0, +1}, and with thermal fluctuation the unsettled remainder passes as carry (N5). Corresponded.

## Molecular, chemical and biological

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1952 | Morphogenesis by reaction–diffusion | Turing | Activator-inhibitor diffusion forms patterns | Chemical/ biological systems | link |
| 1969 | Genetic regulatory Boolean networks | Kauffman | Random Boolean nets model gene regulation; attractors as cell types | Software | link |
| 1989 | Reaction–diffusion / BZ computing | Kuhnert, Agladze, Krinsky; Adamatzky | Chemical waves process images, find paths, Voronoi | Belousov–Zhabotinsky media | link |
| 1993 | Molecular logic gates | de Silva | Molecules switch fluorescence as logic outputs | Photoionic molecules | link |
| 2000 | Synthetic gene circuits | Gardner, Collins; Elowitz, Leibler | Engineered toggle switch and repressilator in cells | E. coli | link |
| 2000/2010 | Slime mould computing | Nakagaki, Yamada, Tóth; Adamatzky | Physarum finds shortest maze paths and transport networks | Physarum polycephalum | link |
| 2003/2022 | Wetware / cultured neuron computing | Potter; Kagan et al. (DishBrain) | Living neurons on electrodes learn closed-loop tasks | MEA cultures; Cortical Labs CL1 | link |
| 2006 | DNA origami | Rothemund | Programmable self-assembly (nanofabrication); substrate for molecular computing, not a model of computation | DNA nanostructures | link |
| 2011 | DNA strand-displacement circuits | Qian, Winfree | Seesaw gates compute square roots, neural nets | DNA in solution | link |
| 2018 | Fungal computing | Adamatzky | Mycelial electrical spiking as information processing | Oyster fungi electrodes | link |

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 2019 | Basal cognition / bioelectric computation | Levin | Bioelectric networks compute anatomical goals in tissues | Planaria, Xenopus | link |
| 2020 | Xenobots / computational bio-design | Kriegman, Blackiston, Levin, Bongard | Evolved designs realised as living frog-cell machines | Xenopus cells | link |
| 2023 | Organoid intelligence | Smirnova, Hartung et al. | Brain organoids as biocomputing substrate | Organoids on MEAs (FinalSpark) | link |

In the net: strand displacement and base pairing are meeting closures (complementary pairing, carry 0); a gene-circuit attractor or a Boolean-network attractor is closure as a cell type. Reaction–diffusion patterns and slime-mould networks keep their first passages as memory (N3), and a ring oscillator such as the repressilator matches the ring closure (x, x, x, carry ±1). Corresponded.

**Neuromorphic, in-memory, spintronic, superconducting**

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1971/2008 | Memristor | Chua; Strukov, Williams (HP) | Resistance depends on charge history; fourth circuit element | TiO2 crossbars | link |
| 1987–91 | Rapid single flux quantum logic | Likharev, Semenov | Superconducting flux-quantum pulses encode bits | Josephson junction circuits | link |
| 1988/1990 | Spintronic computing | Fert, Grünberg (GMR); Datta–Das | Electron spin carries and processes information | MRAM, MTJs | link |
| 1989 | Neuromorphic engineering | Mead | Subthreshold analog VLSI emulates neural physics | Silicon retina, cochlea | link |
| 1993 | Quantum-dot cellular automata | Lent, Tougaw | Electron positions in coupled dots carry logic | Metal-island QCA prototypes (Amlani et al. 1999) | link |
| 1999/2020 | Oscillatory neural networks | Hoppensteadt, Izhikevich; Csaba, Porod | Phase-coupled oscillators compute by synchronisation | VO2 oscillators, CMOS | link |
| 2010 | Magnonic / spin-wave computing | Kruglyak, Chumak et al. | Spin-wave interference performs logic | YIG waveguides | link |
| 2010 | BrainScaleS | Meier, Schemmel | Accelerated analog wafer-scale neuron emulation | BrainScaleS-1/2 (Heidelberg) | link |
| 2011/2018 | SpiNNaker | Furber | Million ARM cores route spike packets | SpiNNaker 1/2 (Manchester, Dresden) | link |
| 2014 | TrueNorth | Modha (IBM) | Event-driven chip, 1M neurons, 256M synapses | IBM TrueNorth | link |
| 2015/2018 | Analog in-memory computing | Prezioso, Strukov; Ambrogio, Sebastian (IBM) | Ohm/Kirchhoff laws do matrix-vector products in memory arrays | RRAM crossbars; IBM HERMES PCM chip (2023) | link |

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 2017 | Spintronic oscillator neuromorphics | Torrejon, Grollier et al. | Nano-oscillator dynamics recognise spoken digits | Spin-torque nano-oscillators | link |
| 2018 | Loihi | Davies (Intel) | Spiking chip with on-chip learning rules | Loihi 1/2; Hala Point (2024) | link |
| 2022 | Physical neural networks | Wright, McMahon et al. | Train arbitrary physical systems with physics-aware backprop | Mechanical, optical, electronic systems | link |

In the net: a memristor's state is history as shape (N3); an in-memory crossbar computes where the stored state is read, so the response is a state function (N4). A single flux quantum is a discrete passage that is either present or absent, and spin up/down with an unpolarised state matches {−1, 0, +1}; oscillator networks close by phase locking. Corresponded.

### Reservoir, morphological, amorphous, collective

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1991/2005 | Programmable matter | Toffoli, Margolus; Goldstein (claytronics) | Matter changes physical properties under program control | CAM-8; catoms | link |
| 1991/1992/2001 | Artificial life / Tierra / artificial chemistry | Ray; Fontana; Dittrich | Self-replicating programs and reaction-like rewriting evolve | Software | link |
| 2000 | Amorphous computing | Abelson, Knight, Sussman et al. | Many unreliable locally communicating particles compute collectively | Simulations; Kilobots | link |
| 2002 | Collision-based computing | Adamatzky; Margolus | Logic from collisions of localisations (gliders, solitons) | CA, BZ media | link |
| 2003/2019 | Physical reservoir computing | Fernando, Sojakka; Tanaka et al. (review) | Any rich dynamical physical system serves as reservoir | Bucket of water; photonic, spintronic reservoirs | link |
| 2006/2011 | Morphological computation | Pfeifer, Bongard; Hauser | Body morphology and materials offload control computation | Passive-dynamic walkers; soft robots | link |

In the net: a reservoir is a strand with fading memory (N3) and only the readout closes; in morphological computation part of the computation is held as shape. Amorphous and collision-based computing reach closure through local meetings of signals, and what does not close locally passes as carry to the next scale (N5). Corresponded.

### Multi-valued hardware

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1920/1921 | Many-valued logic | Łukasiewicz; Post | Truth values beyond two | Tunnelling-based ternary CMOS (Jeong, Kim et al. 2019) | link |

Fowler's ternary machine (1840) and Setun (1958) are in earlier catalogues. In the net: three-valued logic is the native alphabet of the net, the balanced set {−1, 0, +1} of Setun, BitNet b1.58 and ternary CMOS. A

window of three trits closes when its sum is 0 mod 3 (N2), so ternary hardware holds the closure test in its own number system. Corresponded.

# Computational theories of physics and life

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 1969 | Calculating Space / digital physics | Zuse | Universe as a cellular automaton | Theory | link |
| 1972/1980 | Autopoiesis | Maturana, Varela | Living systems self-produce their components and boundary | Theory; SCL computational model (Varela 1974) | link |
| 1985/1991 | Anticipatory systems / relational biology | Rosen | Living systems contain predictive models; (M,R)-systems not Turing-simulable | Theory | link |
| 1989 | "It from bit" | Wheeler | Physical reality derives from binary information | Theory | link |
| 1990 | Digital mechanics | Fredkin | Reversible CA underlie physical law | Theory | link |
| 1991 | Enactivism / embodied cognition | Varela, Thompson, Rosch | Cognition as sensorimotor enaction, not representation | Theory; embodied robotics | link |
| 2002 | Computational capacity of the universe | Lloyd | Universe performed ~10^120 operations on ~10^90 bits | Theory | link |
| 2002 | A New Kind of Science | Wolfram | Simple programs generate complexity; computational equivalence | Mathematica | link |
| 2002 | Universal alphabet and rewrite system | Rowlands, Diaz | Zero-conserving rewrite generates an infinite universal alphabet; linked to the nilpotent framework | Theory | link |
| 2013 | Constructor theory | Deutsch, Marletto | Physics as possible and impossible transformations | Theory | link |
| 2014/2016 | Cellular automaton interpretation of QM | 't Hooft | Deterministic CA underlie quantum mechanics | Theory | link |

| Year | Theory or device | Person(s) | Core idea | Device or implementation | Source |
|---|---|---|---|---|---|
| 2020 | Wolfram Physics Project | Wolfram, Gorard | Hypergraph rewriting and multiway causal graphs yield physics | Wolfram Language | link |

In the net: 't Hooft's deterministic cellular automata, the Rowlands–Diaz zero-conserving rewrite system, Rosen's (M,R)-systems, autopoiesis and Wolfram's hypergraph rewriting are the nearest relatives of the net, each with a local rule on one ordered strand, closure as self-maintenance or conservation of zero, and carry between scales. Rowlands–Diaz matches most closely, since the net also starts from 0 and closes back to 0. Corresponded.

### Re-check list

Re-check done (2026-11-01, second judge, every field checked against an opened page). Catalogue 4: 43 items re-checked, the 42 entries marked † plus CLARION's year. 41 confirmed, so their † is removed; 2 corrected: CLARION year, unverified → 1997 (Sun, Neural Networks 1997); BrainScaleS year, 2010s → 2010 (Schemmel et al., ISCAS 2010). Catalogues 1–3: 70 entries whose earlier check rested on search results were also re-checked, and all 70 were confirmed. Still open: none.

## Part XIII — The Vacuum.Net theory of computing

The theory in one sentence: a computation is a strand that keeps only what has not yet closed, passes each closure up one scale as a carry, and therefore needs memory and cost that grow with the number of ternary digits of its work, not with its length. Theorem C1 below proves this for the register; Sun et al. supply the measurement it is tested against.

### XIII.1 The model

| Element | Definition | From |
|---|---|---|
| Strand | A computation is one ordered path of passages σ: I → P | N1 |
| Register | Each passage contributes a trit in {−1, 0, +1}, read by the trit rule relative to the first passage through its node | T1, N3 |
| Closure | A standing (0), a meeting (x, −x) or a ring (x, x, x) closes; nothing else does | N2, T7 |
| Carry | A standing and a meeting close with carry 0; a ring closes with carry x, one passage at the next scale | T7.1, N5 |
| Step | Append a passage at scale 0; while the end of a scale closes, remove it and pass its carry up | T1 digit rule |
| Tension | What stays open at each scale | N2 |

### XIII.2 Theorem C1 (bounded open memory)

*Statement.* Under the step above, after n passages: (i) every scale holds at most two open passages, all equal; (ii) the open passages encode the running sum exactly, $S = \Sigma_k$ (sum at scale k)$\cdot 3^k$; (iii) the number

of occupied scales is at most $\lfloor \log_3 n \rfloor + 1$, so at most $2(\lfloor \log_3 n \rfloor + 1)$ passages are open; (iv) the total cost of reading the open state at every step is $O(n \log n)$, against $n(n + 1)/2$ for reading the full history. Status: proved.

*Proof.* (i) Invariant: each scale holds only nonzero, equal entries, at most two. A pushed 0 is a standing and is removed. A pushed t opposite to the last entry forms a meeting and both are removed. Otherwise t equals the last entry, since the only nonzero trits are ±1; the run grows by one, and at length 3 it is a ring, removed and passed up. (ii) Each removal keeps the value: 0 and $x + (-x)$ add nothing, and three x at scale k equal one x at scale $k + 1$, since $3x \cdot 3^k = x \cdot 3^{k+1}$. (iii) Scale k receives one passage per ring closed at scale $k - 1$, each ring consuming three passages, so scale k receives at most $n/3^k$ passages; it is occupied only if $n/3^k \geq 1$. (iv) Sum (iii) over the steps. ∎

The reduction is balanced-ternary addition: the open state is the address of the work done (Foundations T12), with the extra freedom that a scale may hold two equal trits instead of one.

## XIII.3 Computed check

The reduction was run on streams of $10^3$ to $10^6$ passages; in every run the open state reproduced the running sum exactly.

| Passages n | All windows rings (all +1): open passages | $\log_3 n$ | Read cost per step | Random trits: read cost per step | Full history: read cost per step |
|---|---|---|---|---|---|
| 1,000 | 4 | 6.29 | 5.97 | 2.87 | 500 |
| 10,000 | 8 | 8.38 | 8.03 | 3.65 | 5,000 |
| 100,000 | 10 | 10.48 | 10.19 | 4.86 | 50,000 |
| 1,000,000 | 14 | 12.58 | 12.34 | 5.51 | 500,000 |

For the all-ring stream the read cost per step approaches $\log_3 n$: 5% below it at $10^3$ passages, 2% below it at $10^6$. For random trits it is lower, because meetings cancel: the running sum of a random walk grows as $\sqrt{n}$, so the open state grows as about $\frac{1}{2} \log_3 n$ (5.51 against 6.29 at $10^6$). No scale ever held more than two passages.

### XIII.4 The correlation with Sun et al.

| Finding (Sun et al. 2026) | In the net | Status |
|---|---|---|
| Input cost grows as n^1.92 (GPT-5.6-sol) and n^1.82 (GPT-5.6-luna) in the number of calls | A path that keeps its whole history reads n(n + 1)/ 2 passages: exponent 2. Caching brings it below 2 | Computed from the paper's figures |
| Solved sets are not nested | N3: the first passage fixes the direction; another setting gives another path, not a longer one. T11: two different closed windows do not join without tension | Corresponded |
| Higher effort keeps running on tasks every level solves | The closure test lies outside the path (the evaluator), so the path continues past closure | Corresponded |
| Reasoning off costs more (21.5 against 5.4 calls) | N5: what does not close inside a call passes to the trajectory scale | Corresponded |
| More self-verification, same outcome | T7.1: a check and return is a meeting, closing with carry 0 | Corresponded |
| Ladder policy: lowest effort first, escalate after verified failure | N5: close at the lowest scale first; only what stays open passes up | Corresponded |

### XIII.5 Hypothesis H1 (closure as compression)

An agent whose context holds only its open items and the carries of its closed sub-tasks, instead of its whole history, has input cost $O(n \log n)$ instead of $O(n^2)$ in the number of calls. The exponent measured by Sun et al. ($\approx 1.9$) then falls towards 1.

*Price.* The reduction erases the order of everything that closed. By Landauer (Part VIII) erasure costs at least $kT \ln 2$ per bit; in an agent the price is that a closed sub-task can no longer be inspected step by step. The net keeps the result and loses the route.

*Test.* Run one agent twice on the same task set: with full history, and with each verified sub-task replaced by one line holding its result. Measure the exponent of input cost against calls. H1 predicts $\approx 1.9$ against $\approx 1$.

## Part XIV — All theories side by side

Of the theories in this book, only the net makes closure the stopping rule, keeps memory bounded by the digits of the work, and derives its cost from the same rule. The others each have one or two of these three; none has all three.

| Theory (year) | Primitive notions | Where memory lives | When a run ends | Cost measure | Relation to the net |
|---|---|---|---|---|---|
| Turing machine (1936) | 7: states, tape alphabet, blank, input alphabet, transition table, start, halt | Unbounded tape | A halting state is reached | Steps, tape cells | Tape = strand; no closure rule |
| λ-calculus (1936) | 3 term forms + β-reduction | In the term | No reduction applies (normal form) | Reduction steps | Normal form = fixed point |
| μ-recursion (1936) | Base functions + composition, recursion, μ | Arguments | First value found | Not native | μ = first closing passage |
| Finite automaton (1951–1959) | States, alphabet, transitions, start, accepting states | Current state only | Input ends | None | Everywhere-closed path = 9 states (T8) |
| Cellular automaton (1940s–2004) | Lattice, states, neighbourhood, rule | Every cell | Not defined | Cells × steps | T10 rewrite is a 3-state CA; net adds carry |
| Complexity theory (1965–1973) | Machine + resource bound | Machine | As machine | Time, space | Cost law of C1: O(n log n) read cost |
| Algorithmic information (1960–1975) | Universal machine, program length | Program | Halting | Program length | Comparison metric for the net (III.1) |
| Denotational semantics (1969–1971) | Domains, continuous functions | Environment | Least fixed point | Not native | Fixed point = eigenform (T10) |
| Type theory (1934–2013) | Types, terms, judgements | Context | A term is constructed | Not native | Proof = closed path |
| Petri nets (1962) | Places, transitions, arcs, marking | Tokens | No transition enabled | Firings | Transition = local closure rule |
| CSP / π-calculus (1978–1992) | Processes, events or channels | Trace | Process stops | Not native | Trace = strand; handshake = meeting |
| Reversible computing (1961–1982) | Reversible gates | Whole history kept | As machine | Heat at erasure | Net erases what closed: pays Landauer |

| Theory (year) | Primitive notions | Where memory lives | When a run ends | Cost measure | Relation to the net |
|---|---|---|---|---|---|
| Quantum computing (1980–1996) | Qubits, unitary gates, measurement | State vector | Measurement | Gates, qubits | Nilpotent closure (Physics T1, C3) |
| Closure theories (1958–2007) | Distinction, re-entry, closure | Organisation | Not defined | Not native | Declared ground of N1–N2 |
| Transformer agents (2017–2026) | Weights, attention, context window | Whole context, re-read each call | Stop token or evaluator | Tokens; grows as n^1.8–1.9 | Full-history path; H1 predicts n log n |
| **Vacuum.Net (2026)** | **Ground (trinity) + 5 axioms** | **Open passages only, ≤ 2 per scale** | **Every window closed** | **Digits of the work: O(n log n)** | — |

**What the net shares with all of them.** A run is a path, and a result is a configuration that stops changing. Turing calls it halting, Church a normal form, Scott a fixed point, Petri a dead marking, Rosen closure.

**What only the net adds.** The three are tied by one rule. The rule that ends a run (closure) is the rule that erases its history (T7) and the rule that passes results up (carry). Every other theory states these separately or leaves one out. That is also the Ockham argument of this book: one rule where the others need three.

## Part XV — Devices past and a proposed new device

Every theory found its device when a carrier with the right native operation existed: the gear for the carry, the switch for Boole, the transistor for von Neumann, the GPU for attention. The net's native operation is closure with carry. No device performs it as one operation yet. This part proposes one, the closure register, in three stages from software to native ternary silicon.

## XV.1 Carriers and their native operation

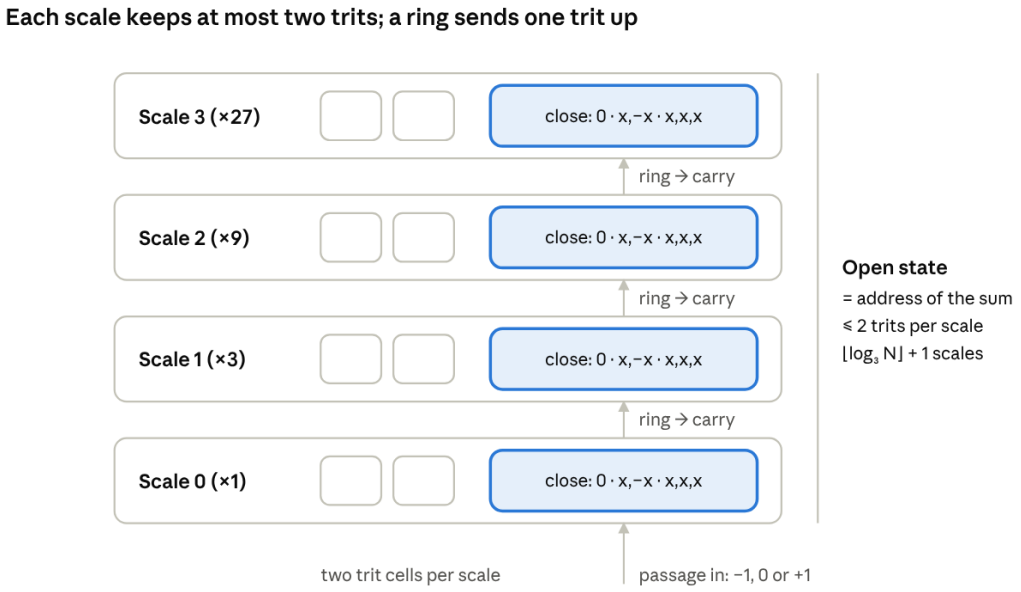
| Period | Carrier | Native operation | Theory it served |
|---|---|---|---|
| 1623–1900 | Gears, wheels | Carry to the next position | Arithmetic |
| 1840, 1958–1970 | Wood; ferrite-core pairs (Setun) | Balanced-ternary digit | Balanced ternary |
| 1937–1945 | Relays, vacuum tubes | Two-state switch | Boolean logic |
| 1948–1960 | Williams tubes, delay lines, ferrite cores | Stored word | Stored program |
| 1960–now | Transistor, CMOS | Two-state switch at scale | All binary theory |
| 2008–now | Memristor | Resistance that keeps its history | Neuromorphic, in-memory computing |
| 2010–now | GPU, TPU | Dense multiply–add | Neural networks, transformers |
| 2019–2025 | Ternary CMOS (UNIST); ternary gate patents | Three-state switch | Ternary logic |

## XV.2 The closure register

*Function.* A stack of scales. Each scale holds at most two equal trits (Theorem C1). Each scale has one detector with three outcomes: a standing (input 0) leaves no trace; a meeting (input opposite to the stored trit) clears it; a ring (third equal trit) clears the scale and sends one trit up the carry line. The register holds the balanced-ternary address of everything that has passed through it.

*Size.* For up to N passages, $\lfloor \log_3 N \rfloor + 1$ scales suffice. For $N = 10^{12}$ that is 26 scales, 52 trit cells. Computed.

*Energy floor.* Each closure erases information. Erasing one trit costs at least $kT \ln 3 = 4.55 \times 10^{-21}$ J at 300 K (Landauer, Part VIII). Computed. A CMOS gate switch today dissipates $10^{-16}$ to $10^{-13}$ J, about $10^{4.5}$ to $10^{7.5}$ times the floor (Ho et al. 2023), so the floor is not the practical limit.




closure register · four scales, carry upward

A passage enters at scale 0. A standing or a meeting leaves nothing; a ring clears its scale and sends one trit up. What remains in the cells is the balanced-ternary address of the sum (Theorem C1).

## XV.3 Three stages

| Stage | Carrier | What it does | Feasible |
|---|---|---|---|
| 1. Software | Agent harness | Context holds open items + carries of verified sub-tasks (H1) | Now; tests H1 directly |
| 2. Digital | FPGA, 2 bits per trit | Closure register as a hardware accumulator: stream in, address out | Now; standard parts |
| 3. Native ternary | Ternary CMOS or memristive three-state cells | One cell per trit; detector and carry as native gates | Research; carriers demonstrated 2019–2025 |

## XV.4 Uses

- *Agent memory.* An agent's context becomes the register: open sub-tasks stay, closed ones leave only their result. H1 predicts input cost falling from ≈ n^1.9 to ≈ n log n in the number of calls.
- *Ternary neural arithmetic.* Products of ternary weights and ternary inputs are themselves trits; summing them is exactly the stream the register reduces.
- *Streaming counters and ledgers.* Credits and debits are meetings; only the net balance and its open digits are kept.

## Part XVI — Open problems

Five questions decide whether the Vacuum.Net theory of computing stands beside the theories of Part XIV or remains a reading of them.

1. **Universality.** Is the register with rewrite R, carry between scales and first-passage memory (N3) Turing-universal? R alone is linear over $\mathbb{Z}_3$ and therefore not (Part IV). The carry is non-linear. A proof would reduce a known universal system, such as Rule 110 or a tag system, to the net.
2. **The trit rule for real traces.** Theorem C1 holds for any trit stream. For an agent the trits must be read from the trace by the trit rule (Foundations T1, IX.5), not assigned by hand. Without that, the correlation of Part XIII stays a correspondence.
3. **H1 in practice.** Does replacing closed sub-tasks by their carries lower the measured exponent from $\approx 1.9$ towards 1 without lowering the solve rate? The price is the erased route (Part XIII.5).
4. **Equal blocks.** Theorem C1 needs no equal blocks; the ladders $\times 2$ and $\times 3$ (Foundations T15) do. How mixed meetings and rings change the cost law is open (Foundations IX.4).
5. **The native carrier.** Can ternary CMOS or memristive cells perform the closure detection of Part XV as one gate, and at what energy above the floor of $4.55 \times 10^{-21}$ J per trit?

## Record of checks and references

All 74 historical claims in Parts I–XII were checked against at least one independent source in four separate checks on 1 November 2026; none was wrong. The 742 catalogue entries (Part XII-A) were judged under the respondent–judge protocol: 796 candidates, eleven judges, 95 corrections, 0 rejections, and a second-judge re-check of 113 entries (111 confirmed, 2 corrected); each catalogue row carries its own source link. Eight of the 74 claims needed a sharper wording, which the text now carries. All numbers in the book were recomputed with code.

## Computed

| Item | What was computed | Result |
|---|---|---|
| Theorem C1 | Closure reduction on streams of $10^3$–$10^6$ trits (all +1 and uniform random) | Open state equals running sum in every run; ≤ 2 passages per scale; read cost per step 5.97–12.34 (all +1) and 2.87–5.51 (random) |
| Input-cost exponent | ln(input-cost ratio) / ln(call ratio) from Sun et al. | 1.92 (GPT-5.6-sol), 1.82 (GPT-5.6-luna) |
| Sign test, corrected SWE-Bench Pro | $2 \times 0.5^8$ | 0.0078 (paper: 0.008) |
| Register size | $\lfloor \log_3 10^{12} \rfloor + 1$ | 26 scales, 52 cells |
| Landauer floor at 300 K | kT ln 2; kT ln 3 | $2.87 \times 10^{-21}$ J; $4.55 \times 10^{-21}$ J |
| Information per window | $\log_2 27$; $\log_2 3$ | 4.75 bits; 1.58 bits |
| Radix economy | 2/ln 2; 3/ln 3 | 2.885; 2.731 |
| Carry non-linearity | c(1) + c(1) against c(1 + 1) | 0 against 1 |

## Wording sharpened after the checks

| Claim | Correction |
|---|---|
| Autopoiesis | Dated 1972/1973 (sources differ) |
| Martin-Löf type theory | 1972 version unpublished until 1998; standard published version 1975 |
| π-calculus | Technical report 1989, journal 1992 |
| Huawei ternary gate | Patent application published March 2025, not granted |
| UNIST 2019 | Funded by Samsung, not Samsung research; third state from tunnelling current |
| Setun | Word length 18 trits; cores in pairs with diodes |
| Erlang | Actor-style, but designed independently of Hewitt |
| Sycamore | 54 qubits on chip, 53 working |

## Sources


- Konstapel, J. (2026), *The Vacuum.Net Theory*, Ninth Edition: Foundations 9.2 and Physics 9.0, Constable Research, Leiden.
- Sun, K., Kwon, Y., Hasan, Z., Zhu, S., Zhou, K., Bianchi, F. (2026), What Effort Level Should I Choose? On the Economics of Reasoning Effort in LLM Agents, arXiv:2610.34491.
- Yang, J. et al. (2026), ARES: Adaptive Reasoning Effort Selection for Efficient LLM Agents.

- Turing, Church, Post, Kleene, 1936: Post, Finite combinatory processes
- Manchester Baby, IEEE Milestone
- Von Neumann universal constructor
- Kleene, RAND RM-704 (1951)
- Hartmanis and Stearns (1965)
- Strachey, Toward a Mathematical Semantics (1971)
- Hewitt, Bishop, Steiger (1973), IJCAI
- Milner, Parrow, Walker, π-calculus report (1989)
- Bérut et al. (2012), Landauer limit
- Arute et al. (2019), Sycamore
- Martin, Odlyzko, Wolfram (1984), additive cellular automata; NKS p. 695, additive rules not universal
- Setun; Brusentsov, Computer Museum
- Ma et al. (2024), BitNet b1.58
- Jeong, Kim et al. (2019), tunnelling-based ternary CMOS
- Huawei ternary patent application (SCMP, 2025)
- Rowlands and Diaz (2002), universal alphabet and rewrite system
- Rosen (1958), relational theory of biological systems
- Kauffman (2005), EigenForm
- Simon (1962), The Architecture of Complexity
- Guo et al. (2024), Ternary Spike
- Ringbauer et al. (2022), trapped-ion qudit processor
- Ho et al. (2023), energy per CMOS switch
- Go FAQ, CSP lineage; occam
- Strukov et al. (2008), The missing memristor found
- Wolfram Physics Project (2020)