ZP-C Illustrated Companion

Why zero is an informational singularity

Information Theory

ZP Companion | Version 2.6 | April 2026

This companion explains the ideas in plain language with diagrams and real-world examples. It is not the formal document — every claim here restates a result already proved in the corresponding technical document. Consult that document for the authoritative mathematics.

What Is ZP-C Doing?

ZP-C establishes that zero is an informational singularity — a point where the cost of describing or reaching it becomes unbounded, no matter how you measure it. Two completely independent measures arrive at the same conclusion from different starting points.

Route 1 — Kolmogorov complexity (algorithmic): How long must a program be to describe a configuration? At the incompressibility threshold P_0, no shorter description exists — the string must be specified in full, every bit. Algorithmic depth is unbounded.

Route 2 — 2-adic surprisal (probabilistic): How much probability mass does a state carry? As a state approaches zero in $\mathbb{Q}_2$, its probability approaches zero, and its surprisal $I(x) = -\log_2 P(x)$ approaches infinity. Informational cost is unbounded.

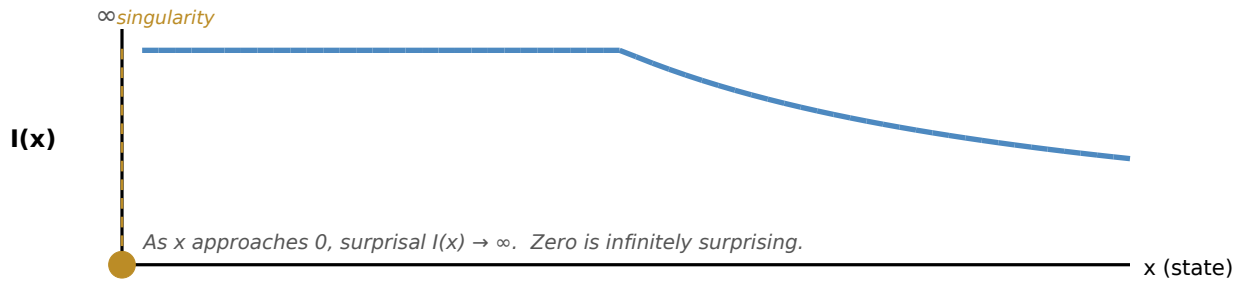
Neither measure knows about the other. K counts program lengths; I(x) counts probability. Both go to infinity at the same threshold. ZP-C operates in $\mathbb{Q}_2$ (from ZP-B) throughout — smooth calculus tools are not valid on a totally disconnected space.

Real-world example — ZIP compression hitting a wall

Some files compress a lot; others barely at all. A file that cannot be compressed further has hit its incompressibility threshold — Route 1. A truly random file also has maximum surprisal: every bit is equally likely, nothing is predictable — Route 2. Both descriptions identify the same extreme: the point where no shortcut exists.

Surprisal: The Cost of a Transition

The surprisal of state x is $I(x) = -\log_2 P(x)$: how surprising it is to observe x. As x approaches 0, P(x) approaches 0, and I(x) approaches infinity. Zero is infinitely surprising — it cannot be reached by any path of finite informational cost.



Surprisal I(x) as x approaches 0 (amber singularity). Every path toward 0 accumulates unbounded informational cost.

Real-world example — A perfectly random password

To describe a truly random password, you have to send every character — you can't summarize it. That's maximum surprisal: the information content equals the full length of the string. Zero in $\mathbb{Q}_2$ is the limit of this — infinite depth, infinite surprisal.

Where the Two Routes Converge

Kolmogorov complexity K and 2-adic surprisal I(x) are genuinely different tools. K measures how long a program must be to reproduce a string — it is a property of algorithms. I(x) measures how rare a state is — it is a property of probability distributions. They share no common definition and were developed in entirely separate branches of mathematics.

Yet both diverge to infinity at the same point: zero. $K(x|n)/n \to 1$ as configurations approach the incompressibility threshold. $I(x) \to \infty$ as states approach 0 in $\mathbb{Q}_2$. The convergence is not engineered — it reflects something structurally real about $\perp$. Kolmogorov complexity and Shannon surprisal were developed in separate branches of mathematics with no shared definition, yet both hit the same barrier at zero. ZP-C uses both as independent confirmation of the same structural fact.

Why the Singularity Forces Execution (L-INF)

The surprisal graph shows that I(x) goes to infinity as x approaches 0. ZP-C makes this precise with Lemma L-INF (Unbounded Surprisal of $\perp$): at the incompressibility threshold P_0, the configuration's surprisal is not just very large — it is formally infinite. The branching measure assigns probability approaching zero to any specific configuration at P_0, so $I(P_0) = \infty$.

This matters because infinite surprisal means no finite external program can bound the informational content of the configuration. A static stored description always has finite content. So a configuration at P_0 cannot be a stored description — it must be something actively running. L-INF is the formal reason P_0 forces execution, and it is what connects the surprisal singularity to the L-RUN argument below.

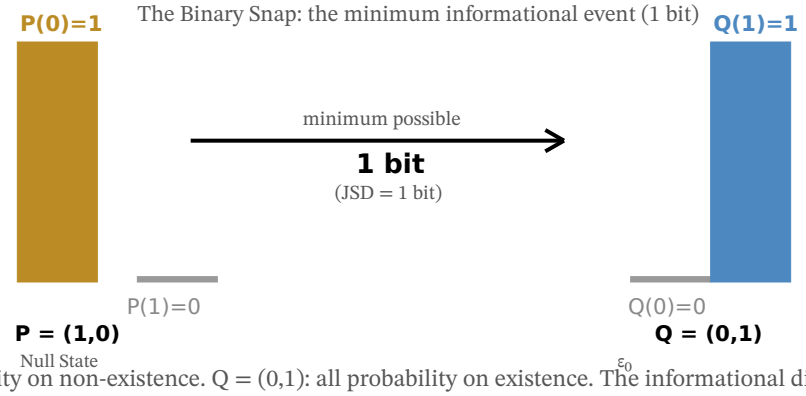
The branching measure — the way ZP-C assigns probabilities to states in D4 — is a representational commitment (RP-2 in ZP-C). It is well-motivated by the binary structure of AX-B1, but it is a choice, not a derivation. The framework labels it explicitly so readers know where a design decision is being made.

Analogy — A file that cannot be described

A truly incompressible file has no pattern — every bit is essential, nothing can be summarized or shortened. At P_0, the configuration is like this file: no shorter description exists. The only way such a thing can "be present" is as something actively running — not a stored record waiting to be read.

The Binary Snap Costs Exactly 1 Bit

The Jensen-Shannon Divergence (JSD) between the zero distribution P = (1,0) and the minimum nonzero state ε_0, Q = (0,1), is exactly 1 bit. These distributions are derived from AX-B1 - not assumed. The Snap is the minimum informational event possible.



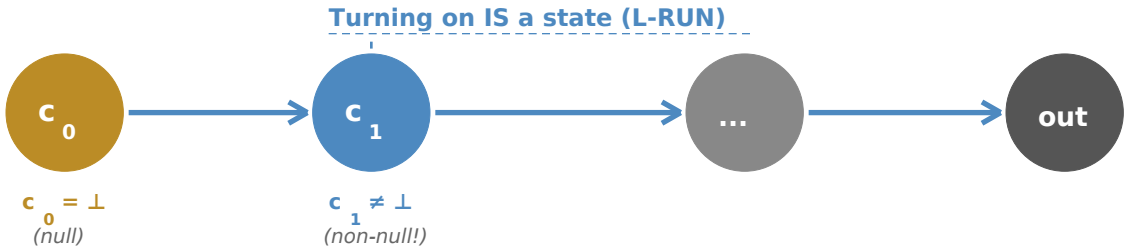
P = (1,0): all probability on non-existence. Q = (0,1): all probability on existence. The informational distance is exactly 1 bit — the minimum possible transition.

Turning On Is a State (L-RUN)

Can a program output ⊥ without passing through any nonzero intermediate state? The answer is no — because the act of turning on is itself a state.

Any program that executes must pass through a "first instruction fetched" configuration (c_1). That configuration is distinct from the not-yet-running state (c_0). By AX-B1 it is non-null. The output being null does not mean the intermediate states were null.

The identification of c_0 with the null state ⊥ is a modeling commitment (CC-2 in ZP-C) — a choice made explicit in the formal document, parallel to CC-1 in ZP-A (which identifies the initial state S_0 with ⊥). It is not derived from the machine definition; it is chosen. Labeling it CC-2 keeps that choice visible.



L-RUN: the transition $c_0 \rightarrow c_1$ is a non-null state change. Execution itself is a state — independent of output.

Real-world example — Booting a computer

When you press the power button, the machine doesn't jump from "off" to "showing your desktop." It passes through BIOS, POST, bootloader, kernel — each a distinct non-null configuration state. Even if programmed to immediately wipe itself and show nothing, it still passed through those intermediate states.

Remember: Computers illustrate the result. L-RUN applies to any Turing machine — the abstract mathematical model of computation. The conclusion is mathematical, not technological.

Two Named Commitments

ZP-C adds explicit labels to two choices the framework makes. Labeling them is not a weakness — it is how the framework keeps track of what is proven versus what is chosen.

CC-2 (Modeling Commitment): The Turing machine initial configuration c_0 is identified with ⊥. This is the same pattern as CC-1 in ZP-A, which identifies the initial state S_0 with ⊥. Neither identification is forced by the definitions — both are deliberate choices that make the multi-layer framework cohere.

RP-2 (Representational Commitment): The branching measure used to assign probabilities to states is a representational choice. It is the natural choice given a binary state space, but it is not the only valid option. Labeling it RP-2 makes the commitment visible to anyone evaluating the framework's assumptions.

Key Result: T-SNAP — The Binary Snap Is a Proven Theorem

The derivation chain: the incompressibility threshold P_0 produces a configuration with infinite surprisal (L-INF). A configuration with infinite surprisal cannot be a static stored description — it must be a live computation (DA-1, ZP-E). Any live computation must execute a first instruction, and that execution is a non-null state (L-RUN). No Turing machine program can produce any output without passing through such a non-null intermediate state. And any non-null state change starting from ⊥ is precisely the Binary Snap (ZP-A D2). ZP-E closes the chain as Theorem T-SNAP. The Binary Snap is no longer assumed — it is derived.