Logical Consistency and Performance Advantages of the Lazy Update World Model: A Thought Experiment Based on Clock Simulation

TAN JUN MING
tttopline8890@gmail.com
https://github.com/junminglazy/Lazy-Update-World-Model

Abstract

The development of traditional virtual worlds has long been constrained by the "impossible triangle" of world scale, content fidelity, and cost. The root cause lies in the O(N) computational complexity of traditional object-centric, per-frame update architectures. To address this challenge, the "Observer-Centric Lazy Update World Model" theory was proposed. This thought experiment, serving as the first proof-of-concept for the theory, aims to verify the internal consistency and theoretical feasibility of its core hypotheses through pure logical deduction and algorithmic complexity analysis.

The experiment designs a virtual scene containing 10,000 clock objects to compare the theoretical performance of a traditional update model (control group) and the lazy update model (experimental group). Clocks were chosen as the ideal experimental subject due to the linear, predictable, and compressible nature of their state changes. The analysis is fundamentally based on the necessary inferences from the theoretical model, rather than empirical observations or actual performance testing.

The predicted results indicate that the lazy update model can reduce computational complexity from O(N) to O(K) (where K is the number of observed objects). In the experimental scenario, performance is expected to stabilize at 60 FPS, whereas the traditional model's performance would drop to approximately 10 FPS. The "scale invariance" demonstrated by this model stems from the principle that the observer's scope (K) does not grow with the total world scale (N). Crucially, for an internal observer, the "compressed evolution" mechanism ensures that the model can, at any moment, provide a result that is mathematically and strictly equivalent to that of a "frame-by-frame update," thereby guaranteeing logical integrity and experiential continuity. Furthermore, the experiment clearly demonstrates that the "dual frame of reference"—experienced by the external observer (who sees the computational optimization) and the internal observer (who experiences a perfect illusion)—is a necessary outcome of the model's dimensional-difference design, not a flaw.

The conclusion of this thought experiment is that the "Lazy Update World Model" is theoretically self-consistent and viable. It proves that by binding computation to the act of observation, the fundamental scale-performance contradiction in virtual world development can be resolved without sacrificing the realism of the internal experience. This provides a solid theoretical direction for building virtual worlds of unprecedented scale.

1. Introduction

The construction of modern virtual worlds has long faced a fundamental dilemma known as the "impossible triangle," which is the difficulty of simultaneously achieving vast world **Scale**, high-fidelity content interaction (**Fidelity**), and manageable development **Cost** within the constraints of limited computational resources. The root of this dilemma is deeply embedded in the "object-centric" architectural paradigm of traditional game engines. This paradigm, by default, executes an update (Update/Tick) call for every active object in the scene on every frame, leading to a system computational complexity that is linearly proportional to the total number of objects (N) in the world, i.e., O(N). As the world scale increases dramatically, this linear growth in computational load inevitably leads to performance bottlenecks, severely limiting the immersiveness and scalability of the virtual world.

To fundamentally break through this architectural shackle, a new theoretical framework has been proposed: the **"Observer-Centric Lazy Update World Model."** This model subverts the traditional computational paradigm by asserting that computational resources should only be allocated to objects perceived by an **Internal Observer**. Its core philosophy is based on the "principle of minimal computation," introducing the concept of a **"Potential State"** where the vast majority of unobserved objects continue to evolve logically, but their computation is deferred. Through two complementary mechanisms—**Law I (Observer Effect and Lazy Update)** and **Law II (Observer Intervention and Causal Chain Settlement)**—the model aims to revolutionize the core logic's computational complexity from O(N) to O(K), where K is the number of observed objects. This value, K, is theoretically independent of the total world scale N, thereby achieving **"scale invariance."**

This paper aims to conduct an in-depth investigation into the theoretical feasibility and internal logical consistency of the "Lazy Update World Model" through a rigorous **Proof-of-Concept Thought Experiment**. This experiment does not seek empirical performance data but rather aims to prove, through pure logical deduction and mathematical analysis, whether the model can indeed deliver an order-of-magnitude performance improvement without sacrificing the temporal continuity and causal integrity experienced by an internal observer.

To this end, an extreme scenario involving 10,000 clock objects was designed to compare and analyze the predicted performance of the traditional update model and the lazy update model. Clocks, due to the deterministic nature of their evolution and the compressibility of their computation, serve as the ideal vehicle for verifying the mathematical equivalence of this theory. Following this introduction, the paper will detail the experimental design and methodology, present the predicted results and phenomenological analysis for both models, conduct an in-depth theoretical analysis and performance evaluation, and finally, in the discussion section, dissect the theoretical foundations of its key phenomena, with the aim of providing a solid theoretical guide to resolving the fundamental contradictions in virtual world development.

2. Experimental Design & Methodology

2.1 Research Method

This study employs the method of a **Proof-of-Concept Thought Experiment**. All experimental procedures, observational phenomena, and performance data are not derived from execution on a physical computer. Instead, they are predicted through rigorous logical deduction and algorithmic complexity analysis based on the theoretical axioms of the "Lazy Update World Model." This method is intended to isolate the core theoretical hypotheses from the confounding factors of specific engineering implementations, focusing solely on verifying their internal logical consistency and feasibility.

2.2 Experimental Environment and Subjects

- **Simulated Environment:** A standard Unity 3D scene is designated as the virtual environment for this thought experiment.
- **Experimental Subjects:** 10,000 clock objects will be instantiated in the scene. This quantity was chosen to theoretically and significantly exceed the performance limits of a traditional single-frame update architecture, thereby magnifying the differences between the two models.

2.2.1 Rationale for Subject Selection

Clocks were selected as the validation subject for this thought experiment because they perfectly satisfy all the characteristics required to verify the lazy update theory:

a) **Predictability of State Change:** The passage of time follows a strict linear law, ensuring that the correct state at any given moment has a unique solution, which facilitates precise logical verification.
b) **Compressibility of Computation:** Its continuous evolution process can be mathematically compressed into a single calculation. This makes it the ideal model for demonstrating the mathematical equivalence between "compressed evolution" and "frame-by-frame updates."
c) **Intuitiveness of Phenomena:** The "rotating" and "static" states of the clock hands are immediately apparent, making it possible to powerfully demonstrate the observational differences between the internal and external perspectives.
d) **Universality of Concept:** Time is one of the most fundamental systems in a virtual world. If the model can be successfully applied to the time system, it demonstrates the universal potential of its methodology.

2.3 Definition of Roles and Observers

In accordance with the core theory, the experiment defines two types of observers with fundamental differences:

- **External Observer:** Defined as us, existing outside the dimension of the virtual world, such as a player or developer. The External Observer possesses a global perspective and can perceive the underlying mechanics of the system. However, their key characteristic is the **inability to trigger world state updates** (lacking the `UpdateStateOnObserve()` function).
- **Internal Observer:** Defined as an entity existing within the dimension of the virtual world, such as a Non-Player Character (NPC). The Internal Observer has a limited, first-person field of view. Their core characteristic is the **ability to trigger world state updates** (possessing the

`UpdateStateOnObserve()` function). Their act of "perception" is the sole trigger that drives an object's transition from a "Potential State" to a "Current State."

2.4 Experimental Groups

To form a rigorous comparison, this thought experiment establishes a control group and an experimental group:

- Control Group - Model A: Traditional Update Model
  This model adheres to the architecture of mainstream game engines. In every frame, the system will iterate through and execute the `Update()` function for all 10,000 clock objects, regardless of whether they are being seen by any observer.
- Experimental Group - Model B: Lazy Update Model
  This model is built entirely upon the "lazy update" theory. By default, clock objects exist in a "Potential State" that consumes no computational resources. Only when a clock is detected within the view frustum of an Internal Observer (the NPC) will the system invoke an `UpdateStateOnObserve()` function on it, updating its state to the current moment via "compressed evolution."

2.5 Key Parameters and Constraints

- **Target Framerate:** Set to 60 FPS, meaning the computational budget per frame is 16.67 milliseconds.
- **Internal Observer's Vision:** The NPC's vision is configured with a 90-degree field of view (FOV) and a 50-meter view distance.

2.6 Evaluation Metrics and Analysis Methods

This experiment will theoretically evaluate the two models from the following three dimensions. All evaluation results are derived from logical deduction and complexity analysis:

a) Performance: Calculate the per-frame CPU time consumption and predict the framerate through computational complexity analysis (O(N) vs. O(K)).
b) Logical Consistency: Verify whether the time readings obtained by the Internal Observer under both models are mathematically and strictly equivalent, using "random spot-check tests" and "continuous observation tests."
c) Observational Phenomena: Describe and predict the visual and experiential differences that are the necessary consequence of the different computational mechanisms, from the perspectives of both the External and Internal Observers.

3. Experimental Procedure

The procedure for this thought experiment is divided into two independent parts, corresponding to the control group (Model A: Traditional Update Model) and the experimental group (Model B: Lazy Update Model). The procedure for each part consists of an initialization phase and an execution & observation phase.

3.1 Procedure for Control Group

3.1.1 Initialization Phase

a) Scene Construction: Construct a standard Unity 3D scene.

b) Object Instantiation: Instantiate 10,000 clock objects, uniformly distributed throughout the scene.
c) Activation of Update Mechanism: Enable the `Update()` function for all 10,000 clock objects. The core logic of this function is to accumulate the current time based on Time.deltaTime in every frame and update the clock's display accordingly.
d) Observer Setup:
- Place an Internal Observer (NPC) in the scene and configure it with a first-person perspective.
- Establish a third-person, global perspective for the External Observer, allowing observation of the entire scene.

3.1.2 Execution and Observation Phase

a) Start Simulation: Begin the scene execution.
b) Perform Navigation: Control the NPC to move through the scene, allowing its line of sight to scan across clocks in different areas.
c) Record Predictions: From the perspectives of both the Internal and External Observers, record the predicted performance data and observational phenomena based on the model's theory for subsequent analysis.

3.2 Procedure for Experimental Group

3.2.1 Initialization Phase

a) Scene Construction: Construct a Unity 3D scene identical to that of the control group.
b) Object Instantiation: Similarly, instantiate 10,000 uniformly distributed clock objects.
c) Set to Potential State: All clock objects are in a "Potential State" by default, meaning their `Update()` function is disabled or non-existent.
d) Implement Observation System:
- Build an observation detection system that continuously tracks the view frustum of the Internal Observer (NPC) (FOV=90°, Distance=50m).
- The document mentions implementing an `UpdateStateOnObserve()` function to determine if any object is currently within the view frustum.

e) Implement Lazy Update Mechanism:
- Write the `UpdateStateOnObserve()` function to serve as the core update logic.
- The mechanism of this function is as follows: when invoked, it first calculates the time difference, `timeElapsed`, between the current world time and the object's last update time, `lastUpdateTime`. Subsequently, based on this `timeElapsed`, it calculates the object's correct current state via "compressed evolution" and updates its display. Finally, it updates the object's `lastUpdateTime` to the current world time.

3.2.2 Execution and Observation Phase

a) Start Simulation: Begin the scene execution.
b) Execute Core Loop: The system's core logic performs the following operations in every frame:
- Invoke the observation detection system to get a list of clock objects currently within the NPC's view frustum.
- Call the `UpdateStateOnObserve()` function only for the clock objects in this list.
- Any clock that was in the field of view in the previous frame but has left it in the current frame will immediately cease to be updated.

c) Perform Navigation and Specific Tests: Control the NPC to move through the scene and execute the following three specific observational tests to comprehensively validate the model's characteristics:
- Random Spot-Check Test: At T=60 seconds into the simulation, control the NPC to randomly and quickly observe 10 clocks in the scene that have not been observed for a long time, to verify the logical correctness of their time.
- Continuous Observation Test: Control the NPC to continuously stare at the same clock for 10 seconds to observe the model's behavior in a continuously activated state.
- Field-of-View Switching Test: Control the NPC to rapidly turn its perspective, quickly shifting its view from a group of clocks in Area A to another group in Area B. This is to verify the transfer of the "activity spotlight" and the instantaneousness of the state transitions.

d) Record Predictions: From the perspectives of both the Internal and External Observers, record the predicted performance data and observational phenomena from all tests, based on the model's theory, for subsequent analysis.

4. Predicted Results & Phenomenological Analysis

This chapter will provide a detailed exposition of the predicted operational results for both the control group (Model A) and the experimental group (Model B), based on the aforementioned experimental design and theoretical model. The analysis will consistently revolve around the two dimensions of the External Observer's global perspective and the Internal Observer's first-person experience.

4.1 Control Group: Traditional Update Model

4.1.1 External Observer's Perspective

From the external, global perspective, all system mechanisms will be transparent. The following phenomena are predicted to be observed:

- **Global Activity:** The hands of all 10,000 clock objects in the scene will be in a state of continuous, synchronized rotation. Regardless of where the Internal Observer (NPC) is located or where it is looking, all clocks will be independently computing and updating.
- **Performance Bottleneck:** As every frame requires processing 10,000 update calls, the system will experience a significant performance bottleneck, manifesting as severe stuttering and a lack of smoothness in the scene.
- **Performance Data:** Performance monitoring tools will display an extremely low frame rate, predicted to be around 10 FPS. CPU utilization will remain consistently high (estimated at 95% or more).

4.1.2 Internal Observer's Experience

For the NPC situated within the virtual world, its experience will be as follows:

- **Logical Correctness:** Despite the poor performance, the time displayed on any clock observed by the NPC at any given moment will be correct.
- **Low-Quality Experience:** When moving through the scene, the NPC's view will suffer from severe screen stutter, and there will be a noticeable delay in operational responsiveness.

4.2 Experimental Group: Lazy Update Model

4.2.1 External Observer's Perspective: The "Activity Spotlight" Phenomenon

The External Observer will witness a "magical" scene that contradicts physical intuition but is in perfect accordance with the model's logic:

- **Large-Scale Stillness:** The vast majority (estimated at over 99%) of the clock hands in the scene will remain completely motionless.
- **Localized Activation:** Only a small number (estimated at around 50-100) of clocks currently within the NPC's field of view will be active, with their hands rotating normally.
- **"Activity Spotlight":** This small, active area will move precisely in tandem with the NPC's line of sight, much like a spotlight. When the NPC's gaze sweeps across a region, the clocks within it will be instantly "awakened"; when the gaze moves away, they will immediately "fall asleep" again.

- **Smooth Performance:** The overall scene will run with extreme smoothness, without any stuttering. Performance monitoring is expected to show a stable 60 FPS frame rate and low CPU utilization (approximately 15-20%).

4.2.2 Internal Observer's Experience: The "Perfect Illusion"

The NPC will be completely unaware of the underlying "lazy" computation mechanism. Its experience will be that of an impeccable "perfect illusion":

- **Absolute Logical Correctness:** A "spot check" on any clock at any moment will reveal the time to be perfectly accurate. For example, if the NPC, at T=00:30, turns back to look at Clock #1, which it last glanced at T=00:00, the clock will correctly display 00:30.
- **Seamless Continuous Perception:** The NPC will not perceive that any clock was ever in a "static" state. Because every act of observation forcibly triggers a precise update to the current time, its perception of the flow of time is perfectly continuous.
- **High-Quality Interaction Experience:** When moving through the scene, the display is smooth, and operational responses are immediate, with no sense of delay.

4.3 Specific Test Scenario Analysis

To further highlight the fundamental differences between the two models, the following is a predictive analysis of specific test scenarios.

Test Scenario	Model A: Traditional Update Model	Model B: Lazy Update Model
Random Spot-Check Test	At T=60s, a randomly checked clock correctly displays the time 00:60. To achieve this result, the clock was **continuously computed approximately 3,600 times** over 60 seconds.	At T=60s, a randomly checked clock also correctly displays the time 00:60. However, to achieve this result, the clock was **computed only once**, at the very instant it was observed.
Field-of-View Switching Test	**External Perspective:** No noticeable change; all clocks continue to rotate. **Internal Perspective:** A smooth transition where all clocks show the correct time, but this is accompanied by stuttering.	**External Perspective:** A distinct transfer of the "activity spotlight" is observed. Clocks in Area A instantly "freeze," while clocks in Area B instantly "jump" from a static state to the correct current time and begin to rotate. **Internal Perspective:** A smooth transition where the time on all clocks is perfectly correct, and the experience is fluid.

Analysis Summary: The predicted results of this thought experiment clearly demonstrate the fundamental differences between the two models. The traditional model, at the cost of immense and continuous computational waste, maintains a world that is objectively and globally active. In contrast, the lazy update model employs a revolutionary computational strategy to create a subjectively perfect and logically self-consistent world for the internal observer at a minimal computational cost. The next chapter will provide an in-depth analysis of the theoretical foundations of these predicted phenomena from the perspectives of algorithmic complexity and mathematical equivalence.

5. Theoretical Analysis & Performance Evaluation

This chapter aims to provide a solid theoretical foundation for the phenomena predicted in the previous

chapter. Through computational complexity analysis and a proof of mathematical equivalence, we will explain from a theoretical level why the lazy update model can achieve revolutionary performance improvements while guaranteeing logical consistency. It must be emphasized that all analysis in this section is based on logical deduction from the theoretical model, not on actual performance test data.

5.1 Computational Complexity Analysis: The Revolution from O(N) to O(K)

5.1.1 Control Group: The Linear Performance Bottleneck of O(N)

In the traditional update model, the system's computational load is directly proportional to the total number of objects (N) in the world. According to the experimental design:

- **Total Number of Objects (N):** 10,000
- **Cost of a Single Update (C):** Assumed to be 0.01ms (a theoretical estimate)
- **Target Frame Budget (T_{budget}):** 16.67ms (corresponding to 60 FPS)

The required CPU time per frame (T_{cpu}) is calculated as follows:

$$T_{cpu} = N \times C = 10,000 \times 0.01ms = 100ms$$

Since the calculated T_{cpu} (100ms) far exceeds the T_{budget}(16.67ms), the system cannot achieve the target frame rate. Its theoretically predicted frame rate (FPS_{pred}) is:

$$FPS_{pred} = 1000ms / T_{cpu} = 1000ms / 100ms = 10FPS$$

This analysis indicates that the performance of the traditional model declines linearly as the total number of objects (N) increases, which is its inherent architectural bottleneck.

5.1.2 Experimental Group: The Scale Invariance of O(K)

In the lazy update model, the computational load is proportional to the number of observed objects (K). The value of K is not an arbitrary variable; it is determined by the physical limitations of the internal observer and environmental factors, making it a theoretically constant value that does not grow with N.

- **Theoretical Derivation of K:** The value of K is primarily limited by the observer's view frustum, view distance, and occluders in the scene. Based on the experimental parameters (90° FOV, 50m view distance) and a conservative estimate of occlusion in a typical scene, the number of objects effectively observed at any given moment (K) is expected to be between 50 and 100.

The required CPU time per frame (T_{cpu}) is calculated as follows (using K=100 as an example):

$$T_{cpu} = K \times C = 100 \times 0.01ms = 1ms$$

Since the calculated T_{cpu}(1ms) is far below the T_{budget} (16.67ms), the system can easily maintain the target frame rate. Its theoretically predicted frame rate will stabilize within the range of 60 FPS. This analysis reveals the core advantage of the lazy update model—**Scale Invariance**: as long as the observation scope K remains stable, the performance cost of core logic computation will remain at the same level, regardless of how the total world scale N grows.

5.2 Mathematical Equivalence Analysis: The Logical Guarantee of Compressed Evolution

The theoretical foundation that allows the lazy update model to provide a "perfect illusion" for the internal observer is the strict mathematical equivalence between "compressed evolution" and "frame-by-frame updates." For a linear evolution system like a clock, its state evolution function can be expressed as `e(state, time) = state + time`.

- **Frame-by-Frame Update (Traditional Model):** After n frames (a total duration of $n \times \Delta t$), the final state, state(n), is obtained through n cumulative calculations:

$$state(n) = state(0) + \Delta t + \Delta t + ... + \Delta t = state(0) + n \times \Delta t$$

This process requires n calculations.

- **Compressed Evolution (Lazy Model):** When the object is first observed after a duration of $n \times \Delta t$, the system performs a one-time calculation using the evolution function e:

$$state(n) = e(state(0), n \times \Delta t) = state(0) + n \times \Delta t$$

This process requires only 1 calculation.

Conclusion: The final states derived from both methods are identical, but their computational costs differ by a factor of n. This proves that "skipping" the unobserved intermediate steps is not a form of cutting corners, but rather an efficient computational strategy based on mathematical equivalence. It is a direct manifestation of the "principle of minimal computation."

5.3 Comprehensive Performance Evaluation

5.3.1 Key Metrics Comparison: The theoretical predicted differences in key performance metrics between the two models are as follows:

Metric	Traditional Model	Lazy Model	Theoretical Ratio
Algorithmic Complexity	O(N)	O(K)	N / K
Objects Updated Per Frame	10,000	~100	100 : 1
CPU Time Per Frame	~100ms	~1ms	100 : 1
Predicted FPS	~10	60 (Stable Range)	1 : 6

5.3.2 Performance Scalability Prediction at Different Scales: The powerful effect of "Scale Invariance":

World Scale (N)	Traditional Model Predicted FPS	Lazy Model Predicted FPS	K Value (Theoretically Constant)
1,000	60	60	~100
10,000	10	60	~100
100,000	1	60	~100
1,000,000	~0.1	60	~100

Analysis Summary: The theoretical analysis and performance evaluation clearly demonstrate that the lazy update model is logically rigorous and revolutionary in its performance. By shifting the computational paradigm from "existence is computation" to "perception is computation," it successfully decouples performance from world scale, providing a solid theoretical basis for building ultra-large-scale virtual worlds.

6. Discussion

The predicted results and theoretical analysis of this thought experiment clearly reveal the disruptive potential of the "Lazy Update World Model" compared to traditional architectures. This chapter aims to delve deeper into the core theories behind these predicted results, respond to the fundamental questions raised in the introduction, and discuss the limitations of this experiment and its profound implications for solving the "impossible triangle" dilemma.

6.1 Theoretical Elucidation of Core Findings

6.1.1 The Inevitability of the "Dual Perspective": Dimensional Difference and Observational Authority

Undoubtedly, the most striking prediction of this thought experiment is the stark contrast between the "activity spotlight" seen by the external observer and the "perfect illusion" experienced by the internal observer. This "dual perspective" phenomenon is not a design flaw but a necessary consequence of the model's intrinsic design, rooted in the theory of dimensional difference.

- **Fundamental Difference in Observational Ability:** The model explicitly distinguishes the disparate "authority" of the two types of observers. The external observer (player/developer) exists in the real-world dimension and lacks the core function, `UpdateStateOnObserve()`, which is required to trigger state updates in the virtual world. Therefore, their act of "watching" is merely a passive reading of data, allowing them to see only the uncalculated, potential state of objects.
- **Observation as a Trigger:** Conversely, the internal observer (NPC) exists within the virtual dimension, and its perceptual system is deeply integrated with the `UpdateStateOnObserve()` function. Every act of its "perception" is an active, world-altering event that forces the target object to perform a historical reconstruction, transitioning from its "Potential State" to a "Current State" that is consistent with the current time.

This mechanism is the core manifestation of **Law I (Observer Effect and Lazy Update)**. Thus, the emergence of the "dual perspective" is a result of the unequal, asymmetrical authority to change the world's state, which is granted to the internal and external observers existing in different dimensions.

6.1.2 The Cornerstone of the "Perfect Illusion": Mathematical Equivalence and Relative Perception

The reason the internal observer is unable to perceive the underlying computational "jumps" is that the model guarantees the absolute continuity of its subjective experience. This is based on the principle of the relativity of continuous perception.

The internal observer's chain of experience is composed of a series of discrete yet absolutely correct "observation points." Regardless of how long the interval between two observation points is, when the second observation occurs, the "compressed evolution" mechanism will, based on mathematical equivalence, provide a final state that is identical to the result of a frame-by-frame calculation. Since

the NPC can only ever obtain information at the "instant of observation," and this information is always correct, the understanding of the world it constructs is necessarily continuous and logically self-consistent.

6.2 Responding to the Fundamental Questions

This thought experiment provides clear answers to the three fundamental questions posed in the introduction:

1. On the necessity of computation: "If a tree falls in a forest and no one is around to see it, does it really need to be computed every frame?" The conclusion of this experiment is:
   No. It only needs to be able to provide a logically correct result at the moment it is potentially observed.
2. On process versus result: "If the final result is mathematically identical, why choose expensive process simulation over inexpensive result calculation?"
   This experiment powerfully demonstrates that, for the internal experience of a virtual world, the correctness of the result is far more important than the continuous simulation of the process. This is the essence of the "principle of minimal computation."
3. On the definition of realism: "Is the 'realism' of a virtual world its objective, continuous operation, or the subjective, correct experience?"
   This model advocates that the "realism" of a virtual world should be defined as a verifiable, subjectively and logically self-consistent experience, rather than a cost-agnostic, objective replication of real-world physical processes.

6.3 Limitations of This Thought Experiment

As a highly focused proof-of-concept, this thought experiment also has the following limitations:

- **Idealization of the experimental subject:** The clock is a perfect linear evolution system, making the design of its "compressed evolution" function simple. This experiment did not address the immense challenge of designing deterministic evolution functions for complex non-linear, multi-body interactive, or chaotic systems.
- **Partial validation of the core laws:** The design of this experiment perfectly validates **Law I (passive observation)** but does not involve the mechanisms of **Law II (active intervention and causal prediction)** at all. A complete theoretical validation would require designing scenarios such as a "thought experiment with billiard balls" to test the model's ability to handle causal chain settlement in active interactions.
- **Simplification of inter-object dependencies:** The 10,000 clocks in the experiment are mutually independent. Objects in the real world often have complex dependencies, and this experiment did not explore the mechanism for handling a situation where an object in a potential state needs to be queried by another active object.

6.4 Implications for the "Impossible Triangle"

Despite the limitations mentioned above, this thought experiment clearly demonstrates, on a theoretical level, the viability of the "Lazy Update World Model" as a path to solving the "impossible triangle" dilemma. By decoupling computational complexity from the total world scale N, the model makes the infinite expansion of **Scale** theoretically possible. Simultaneously, the vast CPU resources saved can be invested into the high-fidelity physics and AI simulations of the few observed objects, thereby enhancing content **Fidelity**. Ultimately, this extreme optimization of computational efficiency will directly translate into a significant reduction in development and hardware **Cost**. What this model reveals is a revolutionary path that fundamentally breaks through resource limitations by re-architecting the computational paradigm.

7. Conclusion

This thought experiment was designed to serve as a proof-of-concept for the core theories of the "Observer-Centric Lazy Update World Model" through rigorous logical deduction. By comparing the theoretical performance of the traditional update model and the lazy update model in an extreme scenario involving 10,000 objects, we have arrived at clear and instructive conclusions.

The predicted results of the experiment strongly support the core hypotheses of this research. Firstly, by drastically reducing the computational complexity from O(N), which is tied to the total world scale, to O(K), which is related only to the number of observed objects, the lazy update model demonstrates the potential for a theoretical performance increase of a hundredfold or even more, establishing "scale invariance" as a core advantage. Secondly, through its "compressed evolution" mechanism, the model guarantees that the state results provided to the internal observer are mathematically and strictly equivalent to those of a traditional "frame-by-frame update." This ensures absolute logical integrity and continuity without sacrificing subjective experience. Finally, the revealed difference in perspective between the internal and external observers is proven to be a necessary inference of the model's dimensional difference theory—a self-consistent system characteristic rather than a design flaw.

In summary, this thought experiment has successfully validated, on a theoretical level, that the "Lazy Update World Model" is a logically self-consistent, causally complete, and highly promising futuristic framework. It profoundly responds to the fundamental problems in virtual world development, with its core contribution being a paradigm shift in perspective:

The ultimate purpose of computation is to provide a correct and unerring result for the act of observation, not to simulate every unperceived process at any cost.

This thought experiment illuminates a clear and viable theoretical path to breaking through the "impossible triangle" that has long plagued the industry. However, it has certain limitations (such as validating only Law I). It is necessary to extend this thought experiment to more complex interactive scenarios to validate Law II, in order to verify the overall effectiveness and correctness of the observer-centric virtual world architecture: the Lazy Update World Model.