OCP-TAP Episode 4

Timeout and Retry (TAR) in Distributed Systems

Failures cascade into timeout storms (networks) and reconstruction storms (storage), resulting in cascade failures, metastable failures, limpware and grey failures, even in small clusters

[Paul Borrill](#)
paul.borrill@icloud.com

Twitter: [@plborrill](#)

02024-Dec-04 - Version 1.1

DÆ

Take Aways

Insights & Intuition & Results

- Instants are meaningless, only intervals (*on the same computer/timeline*) are relevant
- Photons don't carry timestamps, but timestamps are carried by photons
- The speed of light is the "pivot" around which time and space evolve
- Timeout and retry (TAR) *on different timelines* will silently corrupt data structures
- Shannon entropy is a logarithm. The logarithm of zero is minus infinity.
- Bayesian approaches require a *prior*, which can be unbounded (zero to ∞).
  - Actually, it's much worse: can be $-\infty, -1, -0, +0, +1, +\infty$. We can't do Bayesian statistics under those conditions, mathematically, their results are undefined
- Shannon Entropy is uncertainty, and the same problem applies when you apply the set $-\infty, -1, -0, +0, +1, +\infty$ to Information and Entropy $p_i log(p_i)$

Measurements "appear" instantaneous because there is no background of time on which to measure anything. Timestamps don't help with causal order

Distributed Systems

- Safety (nothing bad will happen)
- Liveness (something good will eventually happen)
  - When packets get Dropped, REordered, Duplicated or Delayed, software has to intervene to "fix" an *impossible problem*
    - Dropped packets destroy causal order determinacy
    - Reordered packets require unbounded reordering buffers
    - Duplicated packets destroy non-idempotent data structures
    - Delayed packets cause timeouts, causing more retries ...

Shannon Information

Entropy is uncertainty

$p_i log(p_i)$

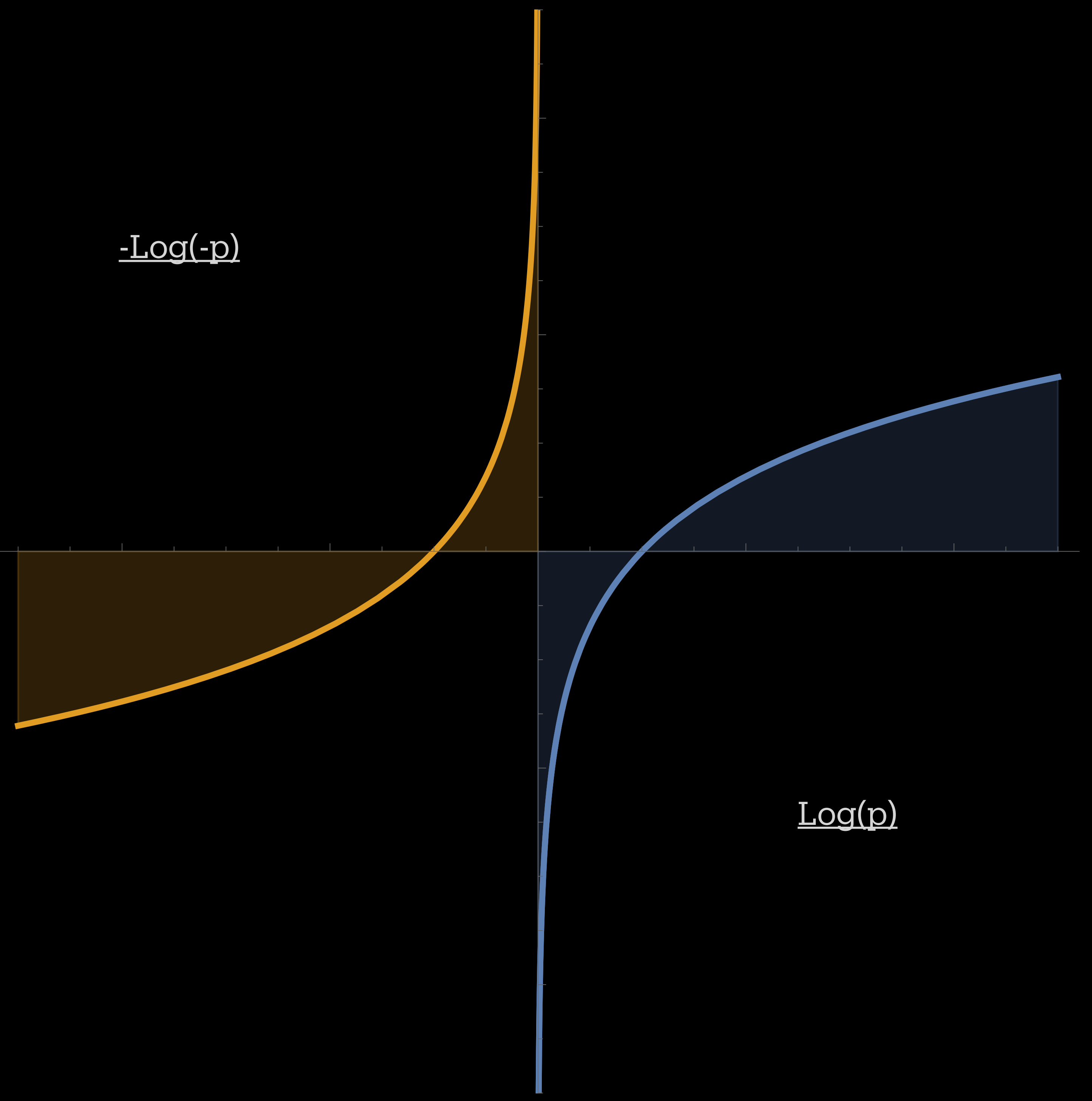
The logarithm of 0 is $-\infty$

In a Shannon Channel, Alice and Bob are exchanging information (through any protocol)

Without a background of time, a photon is an arrival event

An Alternating Bit Protocol (ABP) (with Acks) is the "fastest" possible way to resolve the uncertainty

It will show up as not being able to measure (or depend on) a one-way speed of light



Photons don’t carry Timestamps

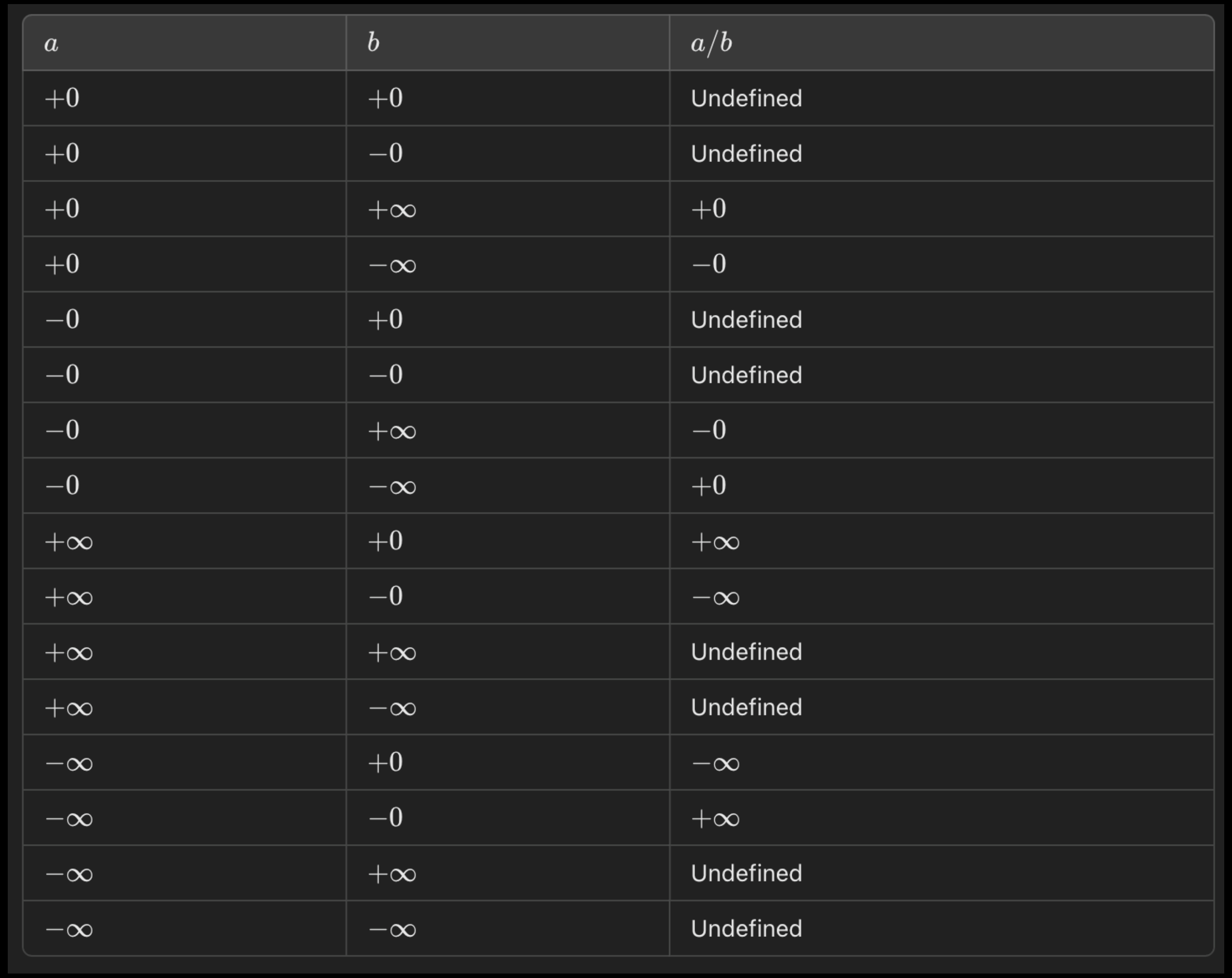
Key problem: Minkowski Error.

An “interval” in Minkowski Spacetime is not the same as an interval measured on a local clock.

$$c = \frac{distance}{time} = \frac{a}{b} = \frac{\infty}{0}$$

We don’t know how far the photon has come,

We don’t know the “time it has been traveling, proper time for a photon = zero !

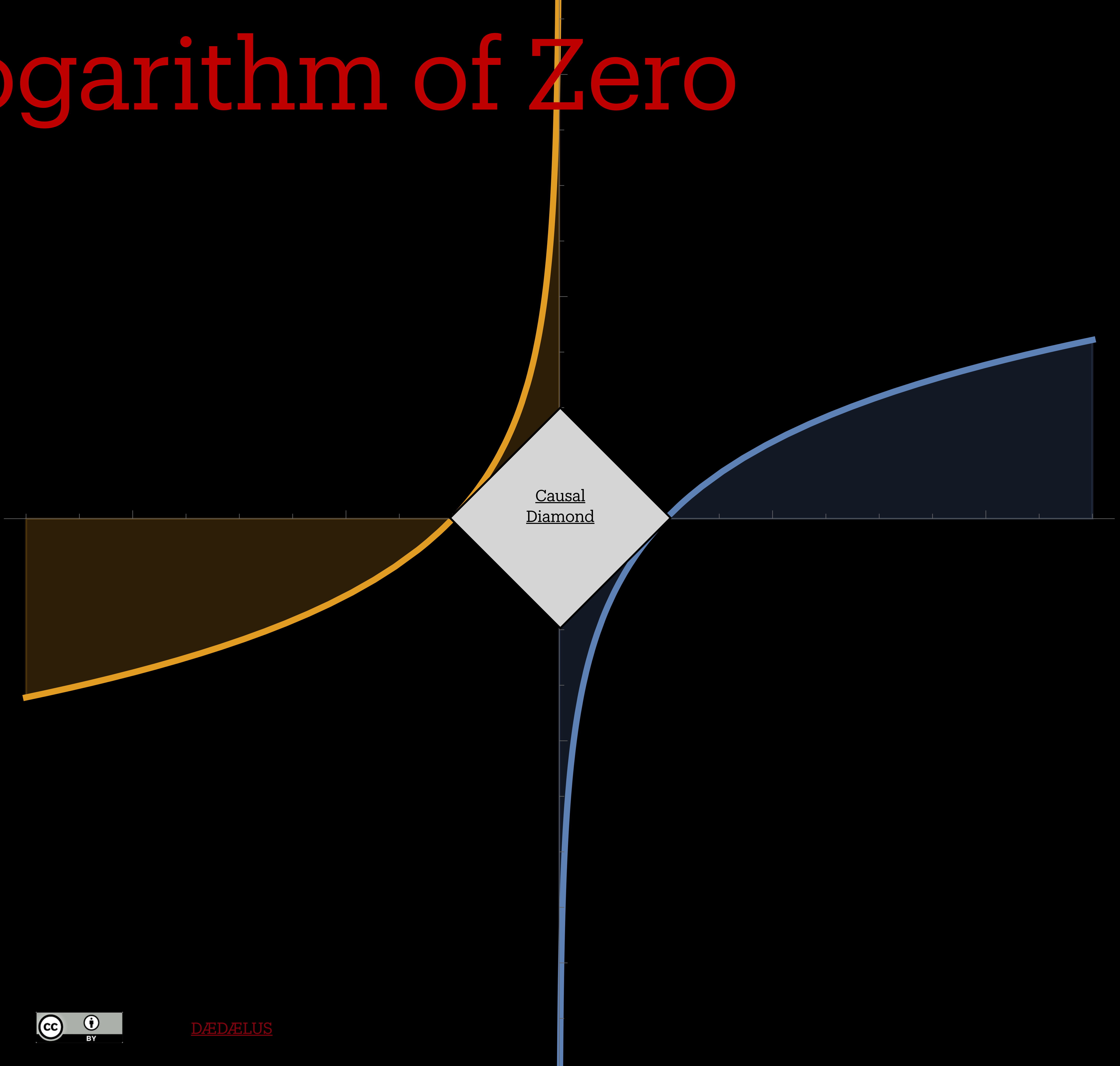


a	b	a/b
$+0$	$+0$	Undefined
$+0$	-0	Undefined
$+0$	$+\infty$	$+0$
$+0$	$-\infty$	-0
-0	$+0$	Undefined
-0	-0	Undefined
-0	$+\infty$	-0
-0	$-\infty$	$+0$
$+\infty$	$+0$	$+\infty$
$+\infty$	-0	$-\infty$
$+\infty$	$+\infty$	Undefined
$+\infty$	$-\infty$	Undefined
$-\infty$	$+0$	$-\infty$
$-\infty$	-0	$+\infty$
$-\infty$	$+\infty$	Undefined
$-\infty$	$-\infty$	Undefined

- There are no “Minkowski manifold coordinates” transmitted (emitted) alongside or with the photon.
- There are no Minkowski manifold coordinates received by the observer (absorber, detector)
- All we have is the arrival of “information” containing wavelength (energy) polarization and phase (we will get to this later)
- We average photons over *intervals* of time defined by the observer, not the emitter
- *It’s impossible to calculate intervals on a Minkowski background, because the coordinate system of the emitter and the observer are inaccessible. They exist only in our imagination.*
- Attempts to recreate a “background” using NTP, PTP, White Rabbit or Sync E may provide “correlations”, but will not “determinism” for causality in our algorithms. Assuming timestamps retain their order prevents us from building reliable distributed systems, i.e., those that don’t silently lose or corrupt data structures

The logarithm of Zero

- No background of time
- The “orientation” of Alice and Bob is symmetric
- Are we “sending” information on a Shannon Channel, or are we receiving information on a return Shannon Channel?
- Does this causal diamond resemble a causal diamond in General Relativity?
- What can we learn from this?



Replacing Causal Diamonds with Causal Squares

Why we can't have nice things in distributed systems

Einstein

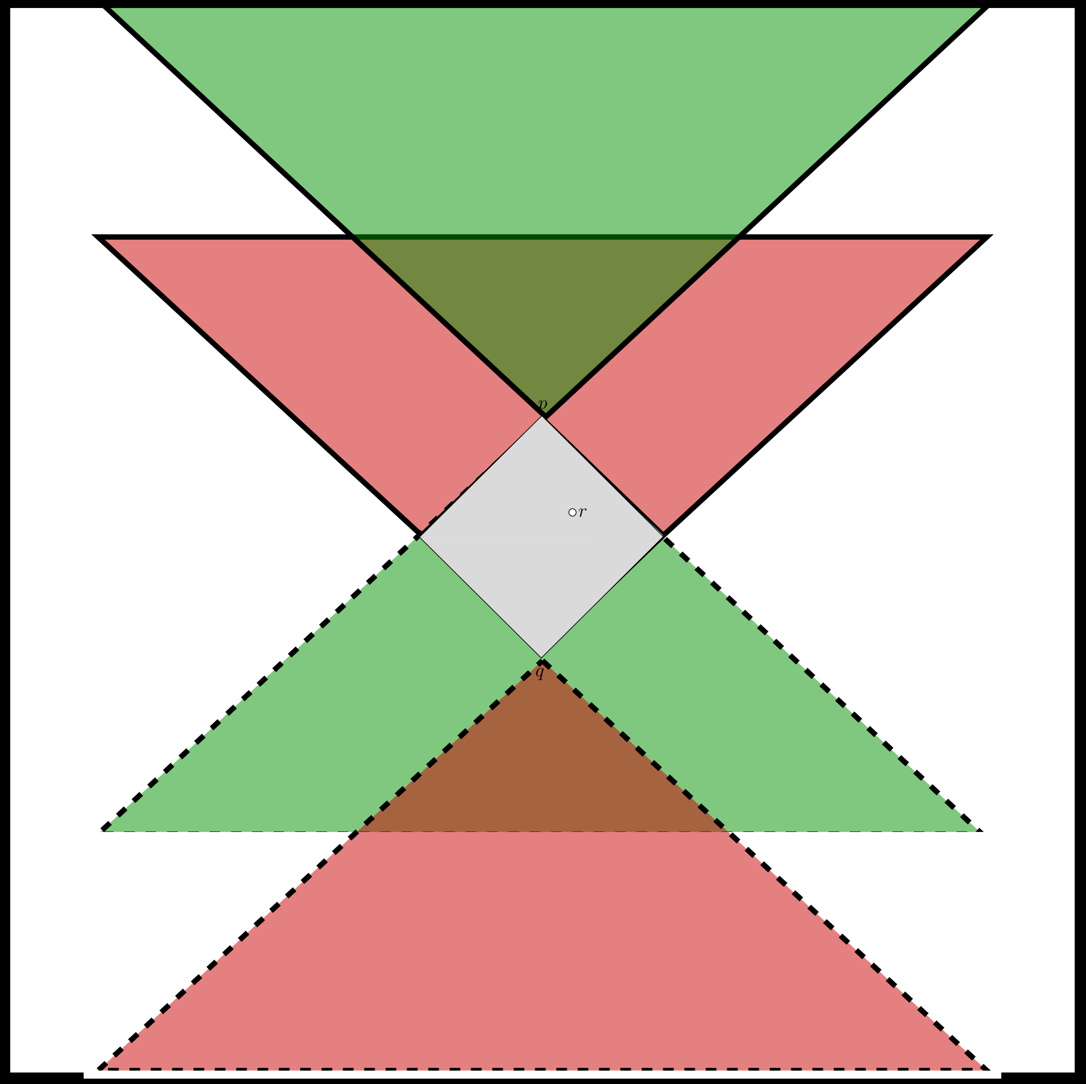
$$\text{velocity} = \frac{\text{light path}}{\text{time interval}}$$

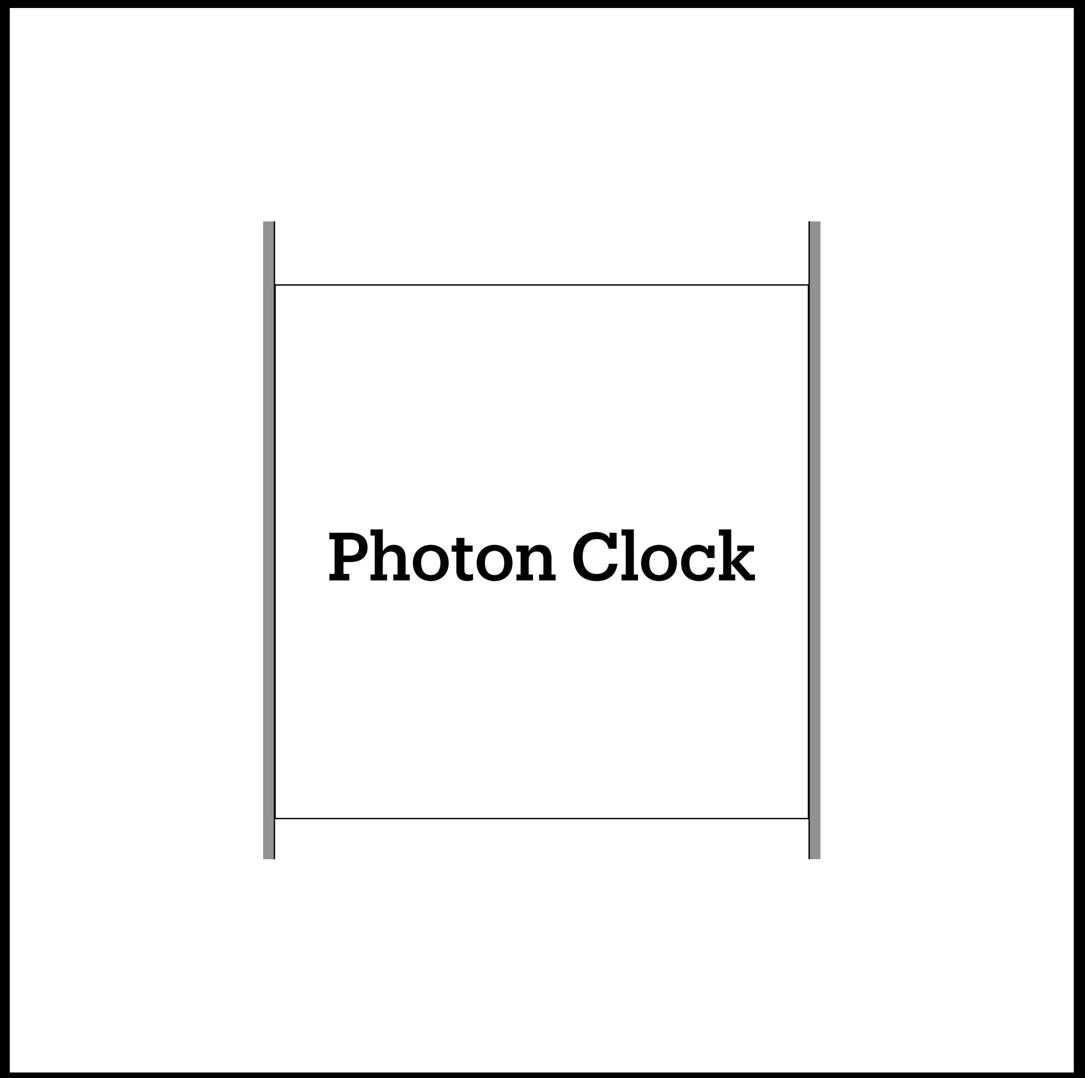
- Problem: How can we assign a meaning to “timestamps” when Einstein’s (special) relativity denies the existence of distant simultaneity?
- The Life of a Photon: Einstein-Shannon Photon Clock
- Photon’s don’t carry timestamps

1. Incoming photon. Uncertainty in distance and time is infinite
2. From Einstein: $velocity = \dfrac{light\ \ path}{time\ \ interval} = \dfrac{\delta(d)}{\delta(t)} = \dfrac{\infty}{\infty}$
3. From Shannon $\dfrac{P(H)}{P(H)} = \dfrac{\log \infty}{\log \infty} = \dfrac{0}{0} = 1$ OR 0 or Undefined?
4. Taking the limit from the (positive) direction yields a different answer than when the limit is taken from the (negative) direction. [See next slide]

Causal Diamond vs. Causal Square

- In General Relativity we use the notion of a causal diamond [3]
- But General Relativity is incompatible with quantum mechanics
- Can we "add" an axiom to GR to reproduce the effects of Quantum Mechanics?
- We will show that the Causal Diamond can be replaced by a causal square
- Results are important to understanding time in distributed systems
- This is why a photon can spend a negative amount of time in an atom cloud [4]





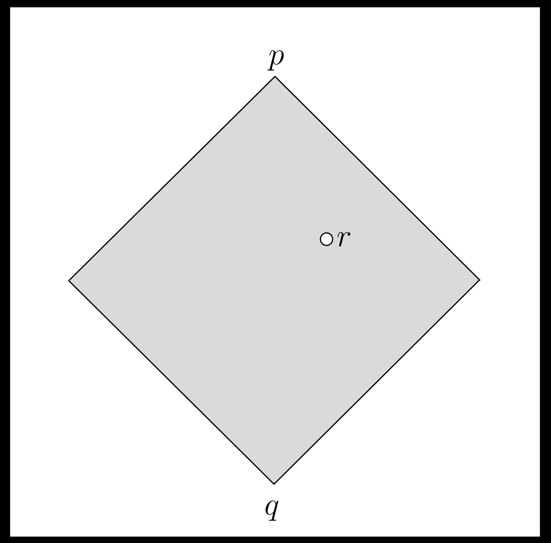


Figure 1: For two spacetime points q, p, with p to the future of q, the causal diamond D_q^p consists of all points that are in the causal future of q and the causal past of p. Sometimes it is of interest to consider a spacetime in which a point r is omitted from the causal diamond, as sketched here.

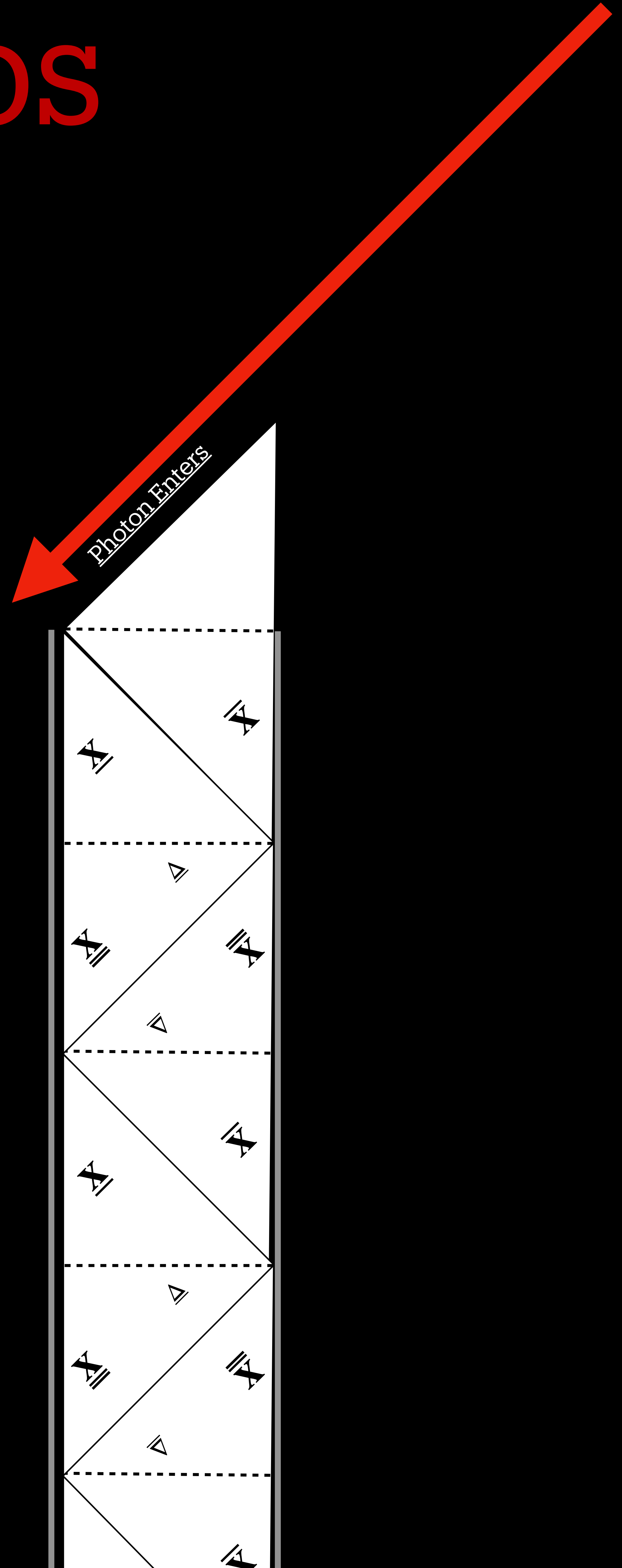
Points in Minkowski Manifold Aren't accessible, not real

[3] Light Rays singularities and all that. Edward Witten arXiv

[4] Experimental evidence that a photon can spend a negative amount of time in an atom cloud [Daniela Angulo et. al.]

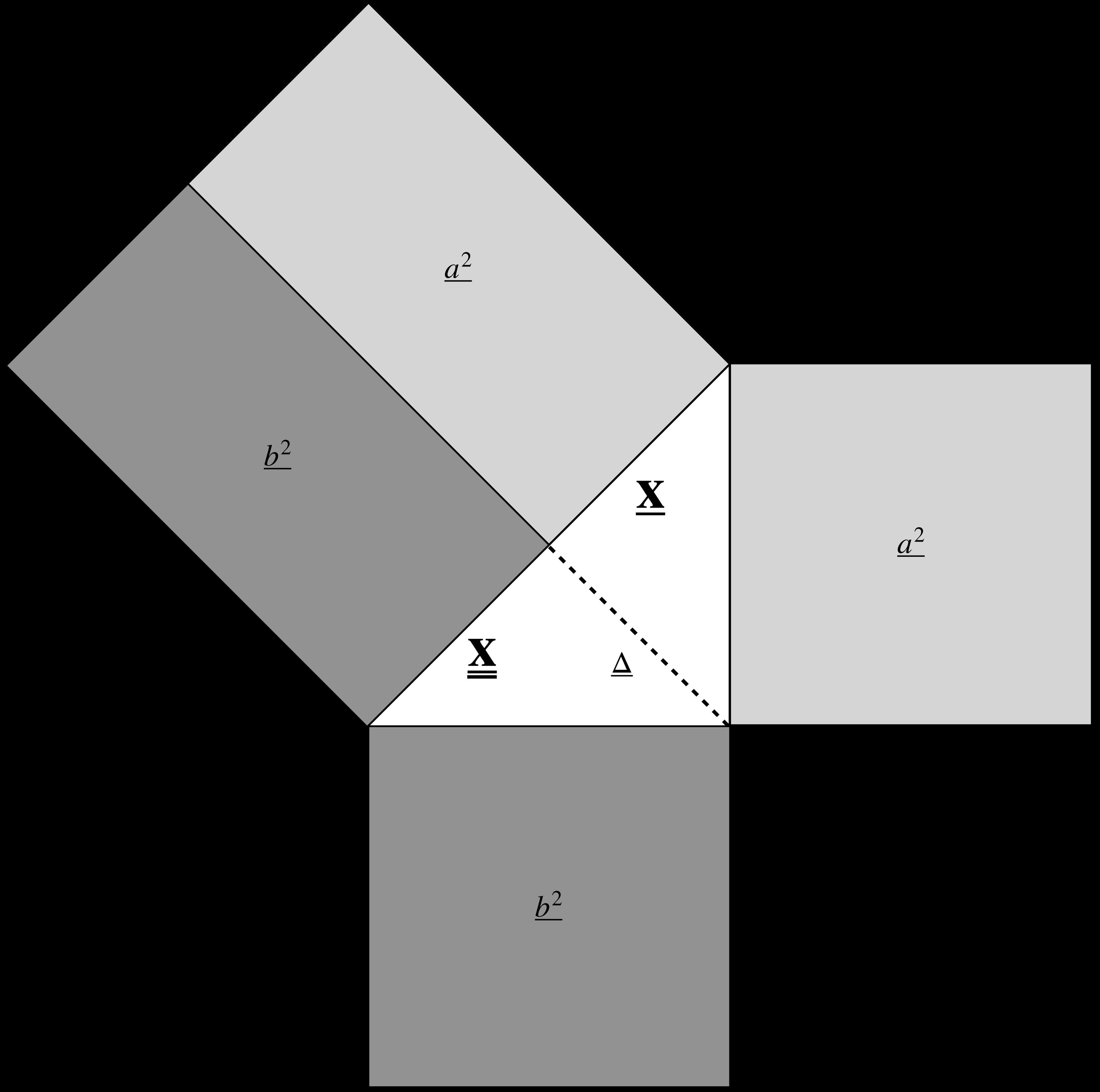
Photons don’t carry timestamps

- Δ is the Distance between Alice and Bob, the initial uncertainty is ∞, because it is unknown how far it has come
- However, when Alice sends the photon on it’s way, it has no idea that Bob is waiting to receive it
- When Bob receives the photon, all he knows is that “someone” like Alice sent the photon, but just like Alice, he has no knowledge of how far it has come.
- However, when Alice receives the photon back, she can measure (with her own clock) the round trip delay. Now Alice knows Δ
- When Bob receives the photon a second time, now Bob knows Delta, if he measures the round trip delay with his own clock. Now Bob Knows Δ.
- Keep on going (in perpetuity) and we have a photon clock, disturb the system and the Hilbert photons go off into space



[1] Inertial Frame -Zero Velocity

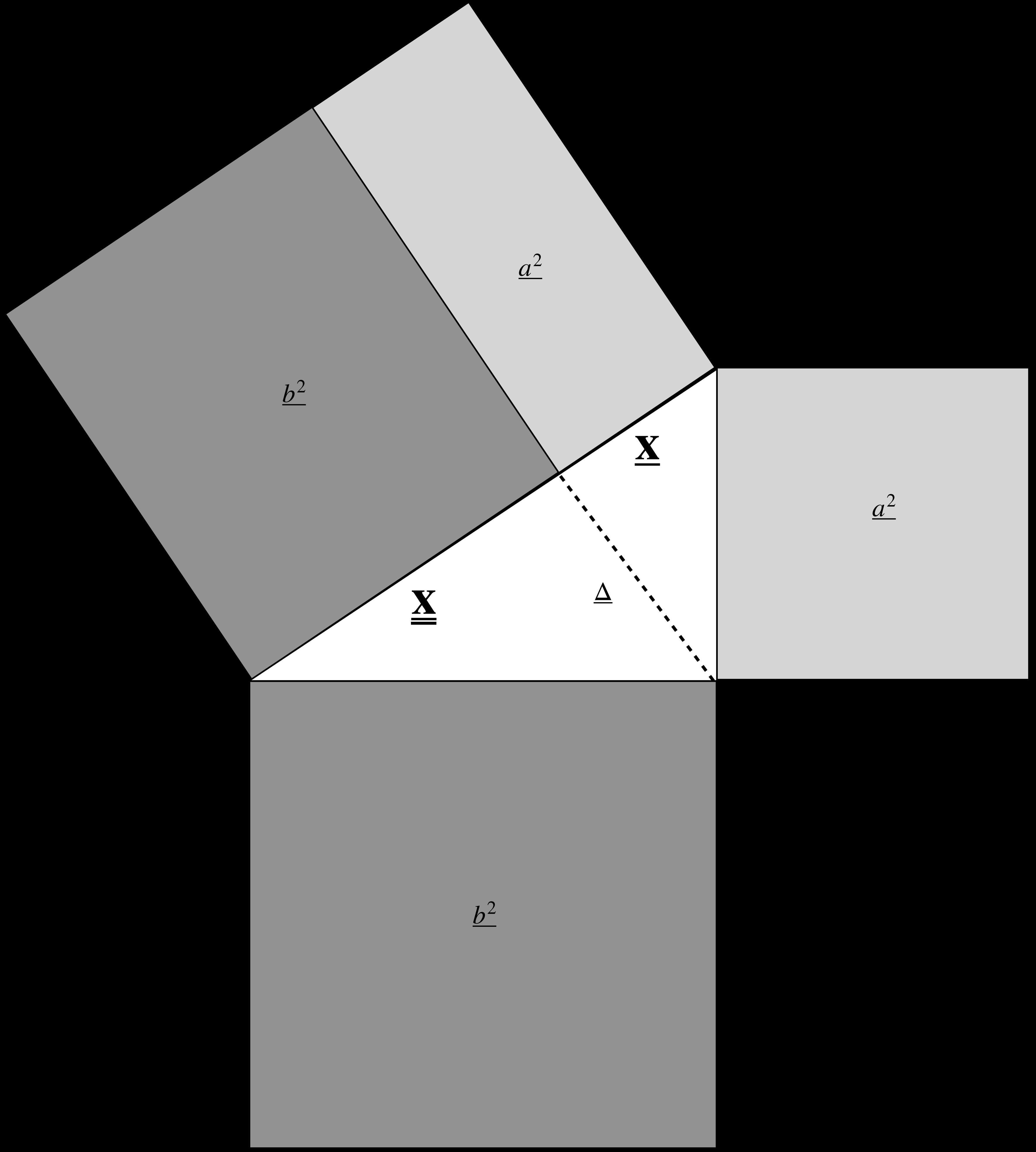
- Einstein: "The introduction of a "luminiferous ether" will prove to be superfluous" [1]
- Paraphrasing: The introduction of a "Minkowski Manifold" will prove to be superfluous
- Begin with Pythagorus, derive Lorentz
- Continue with Barukčič [2]: "Δ denotes the altitude in a right triangle and $\mathbf{x}$ and $\underline{\mathbf{x}}$ denote the segments of the hypotenuse c, of a right angled triangle"
- In an inertial frame we use two similar triangles which added together make an equilateral triangle
- This means $a = b$ and $\mathbf{x} = \underline{\mathbf{x}}$, no distinguishability between the clocks at a and clocks at b (at least not for special relativity)



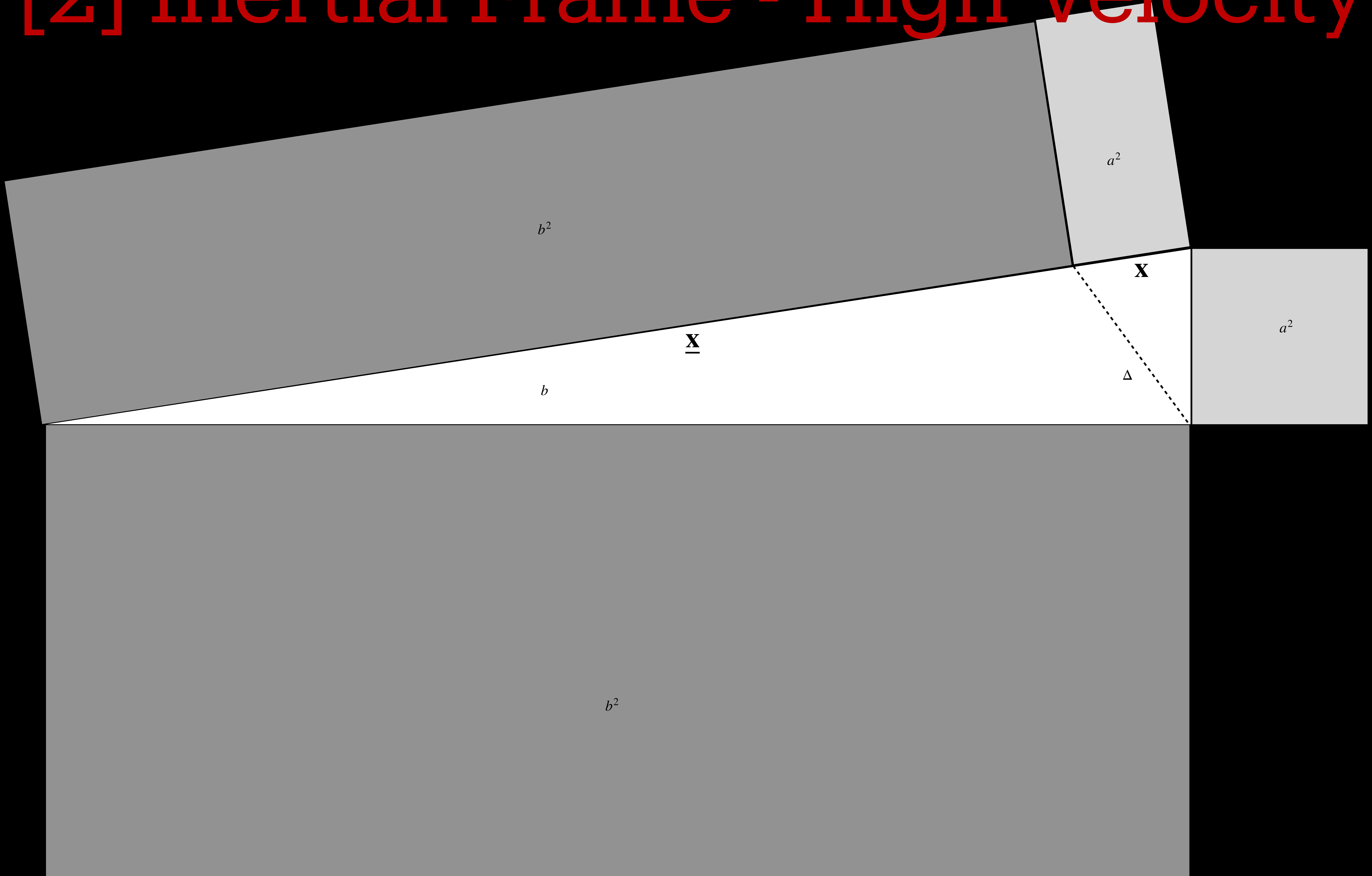
On The Electrodynamics of Moving Bodies by A. Einstein. June30, 1905
[2]

[2] Inertial Frame - Low Velocity

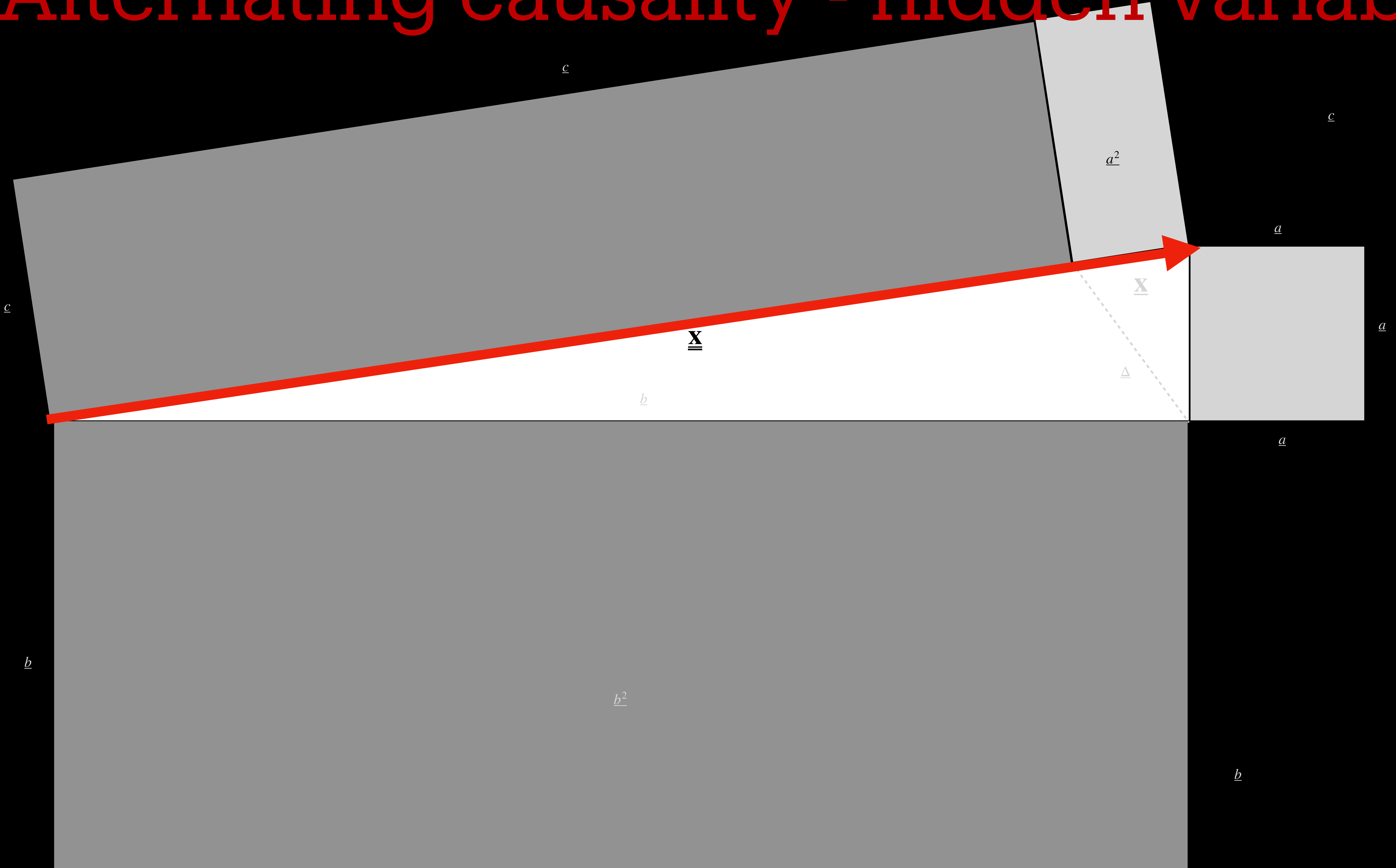
- Continue with Barukčič [2]: "Δ denotes the altitude in a right triangle and **x** and $\underline{\mathbf{x}}$ denote the segments of the hypotenuse c, of a right angled triangle"
- In a non-inertial frame we use two different right angled triangles which added together form Δ, **x**, & $\underline{\mathbf{x}}$ are segments of the hypotenuse.
- $c = \mathbf{x} + \underline{\mathbf{x}}$
- Δ is unknown for an incoming photon
- Photons don't carry timestamps
- It takes the integration of many photons to determine frequency. The Energy of a single incoming photon doesn't tell you how far it's come. It could have been an inch, or from the big bang (13.8B Light years away)



[2] Inertial Frame - High Velocity

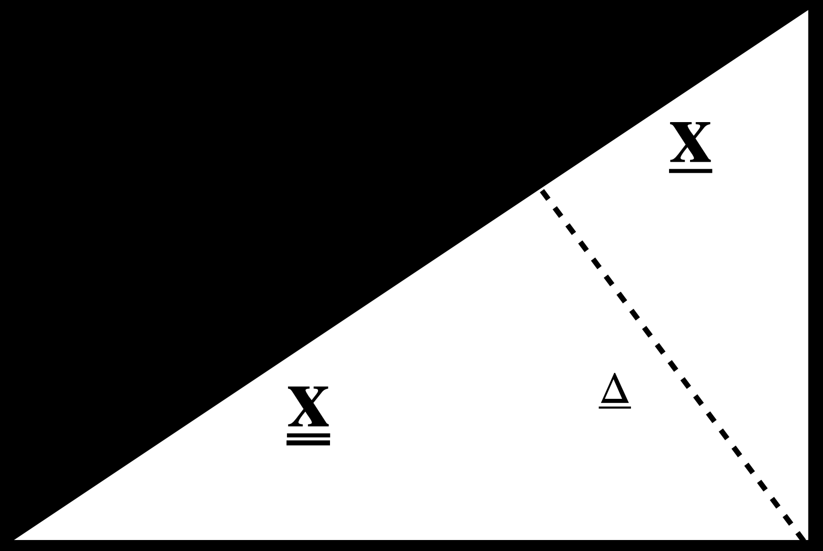


[3] Alternating causality - hidden variables



The Trouble with the GEV

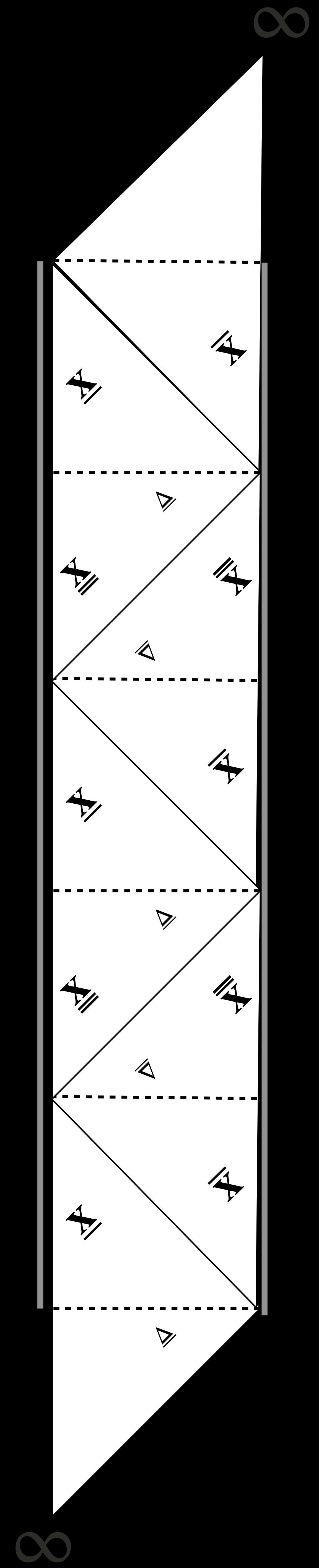
- A God’s Eye View (GEV) misleads us, the universe will appear static from that perspective
- Both General Relativity and Quantum Mechanics impose a human invented “background” of time
- This includes the “direction of time”, which is nothing more than the direction of information transfer. Is it going from Alice to Bob, or from Bob to Alice?
- When it Comes to causality, only a Local Observer View works, where we can account for the internal events that can’t be seen from “outside” - The GEV.
- In the eye of the observer, but who is the observer?
- The photon? — proper time, $\tau = 0$
- The Transmitter?
- The Receiver?



[2] Barukčić, I., 2016. Unified field theory. Journal of Applied Mathematics and Physics, 4(8), pp.1379-1438.

Photons don’t carry timestamps

- Photons bouncing between these mirrors carry Shared (mutual) information between Alice and Bob.
- Until this information is captured (turned into knowledge, or memory), there is no “evolution” that can be called “time”. The photon dynamics is “timeless’ until absorbed.
- When this information is captured, the photon energy is captured by the receiver in memory, and is no longer available to be “shared” in a ping-pong between Alice And Bob
- Einstein’s “hidden variables” lives on, but is hidden in Subtime

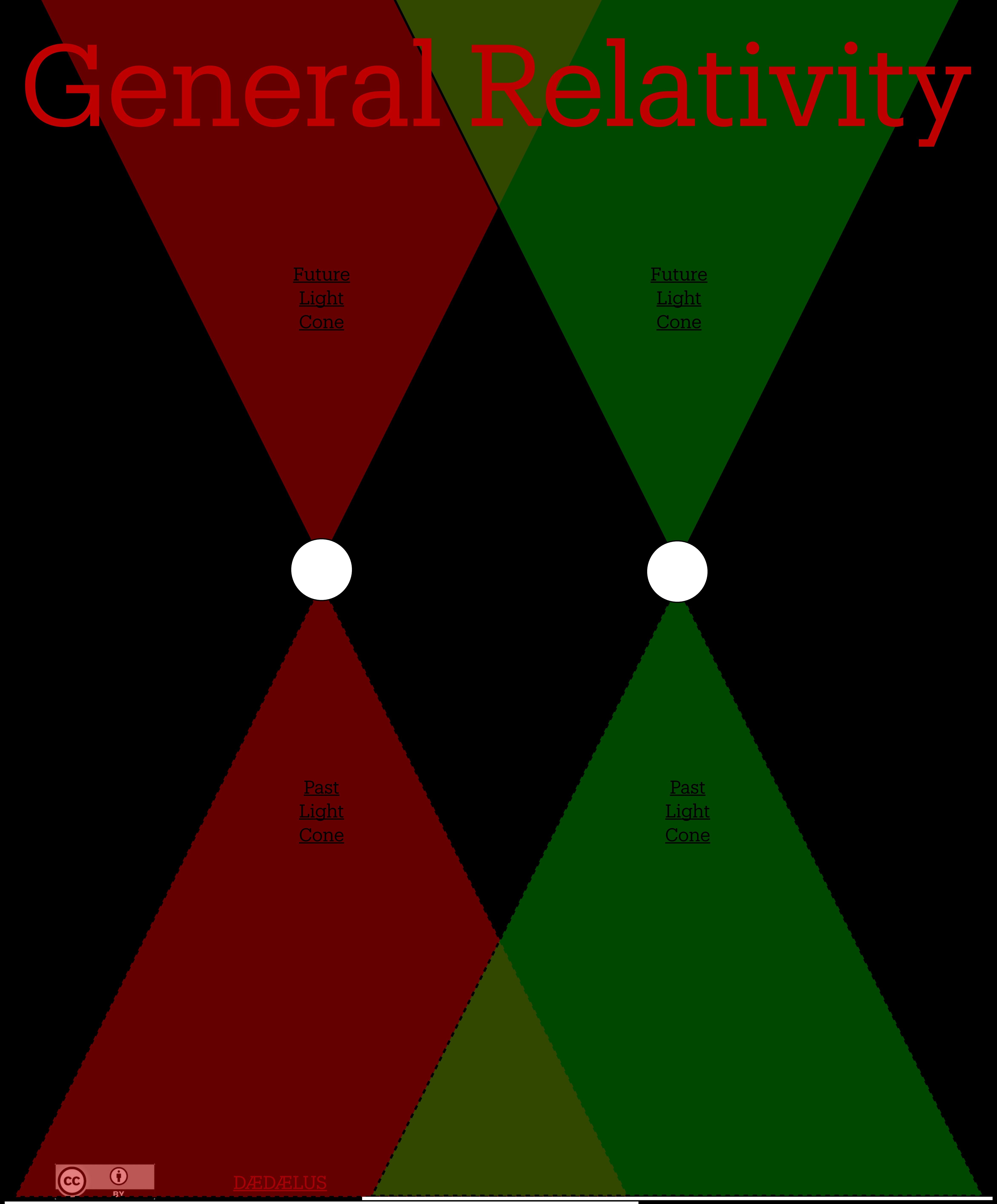


Consequences

- It’s impossible to synchronize clocks in principle, it will therefore be problematic in practice.
  - Clock synchronization error is indistinguishable from (One-Way) latency [Edward Lee
  - The direction of causality is stochastic. See The Life of a Photon Description.
- When Experiments try to measure a one-way latency, such as in Entanglement Experiments, they are committing the Minkowski Error
- When Computer Scientists imagine they can “assume” a background for one-way latency (without , they will find they are not “completions” required by the two-way nature of this theory
- Monotonicity is in the eye of the observer, that’s not how spacetime works.
- No amount of recovery code will save you

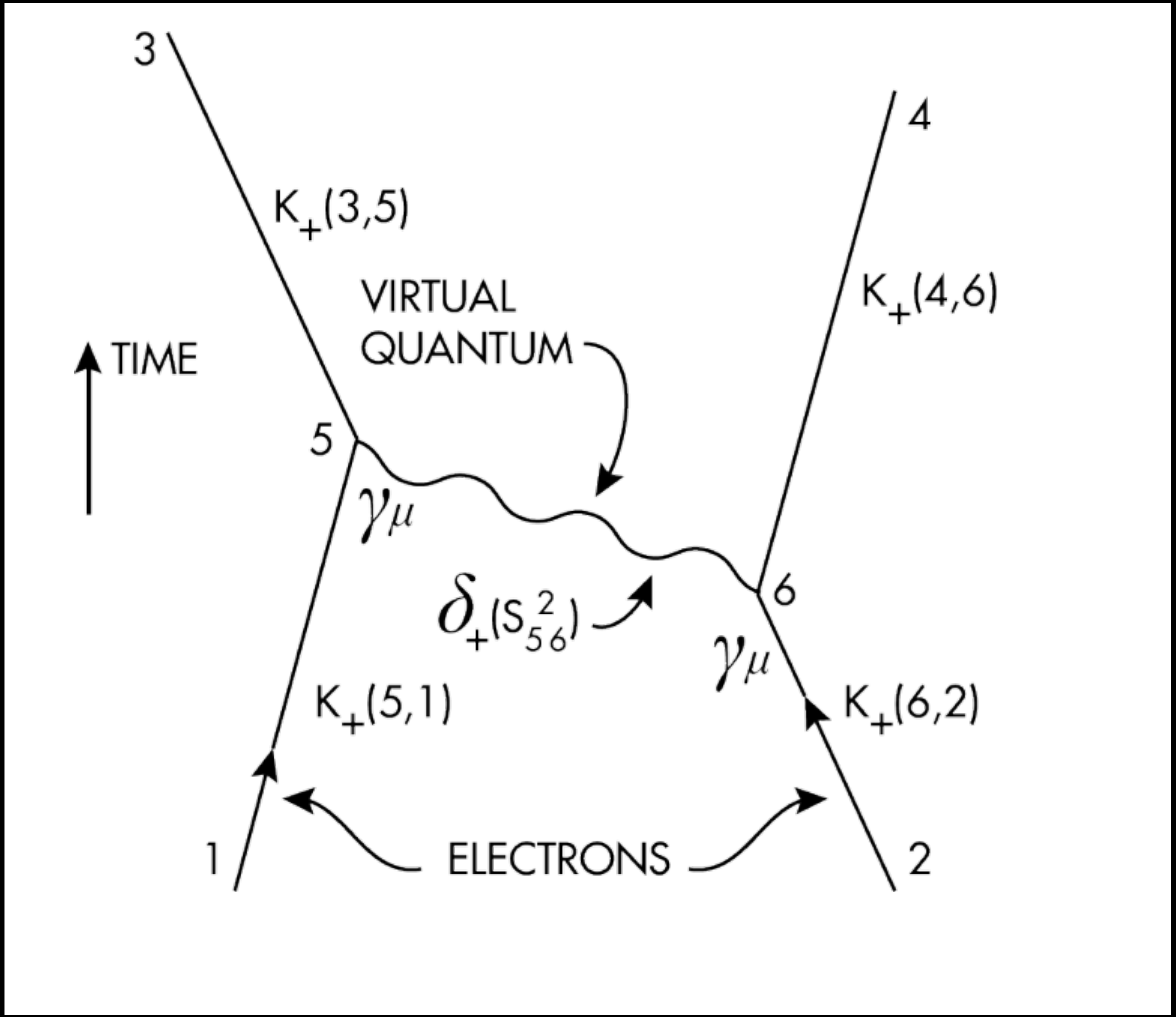
Time in General Relativity

- Time Goes from past to future?
- Causal Diamonds are Symmetric
- This is nonsense.
- We made it up.
- Who decided that the past and the future light cone both point in the same “direction”?



Time in Quantum Theory

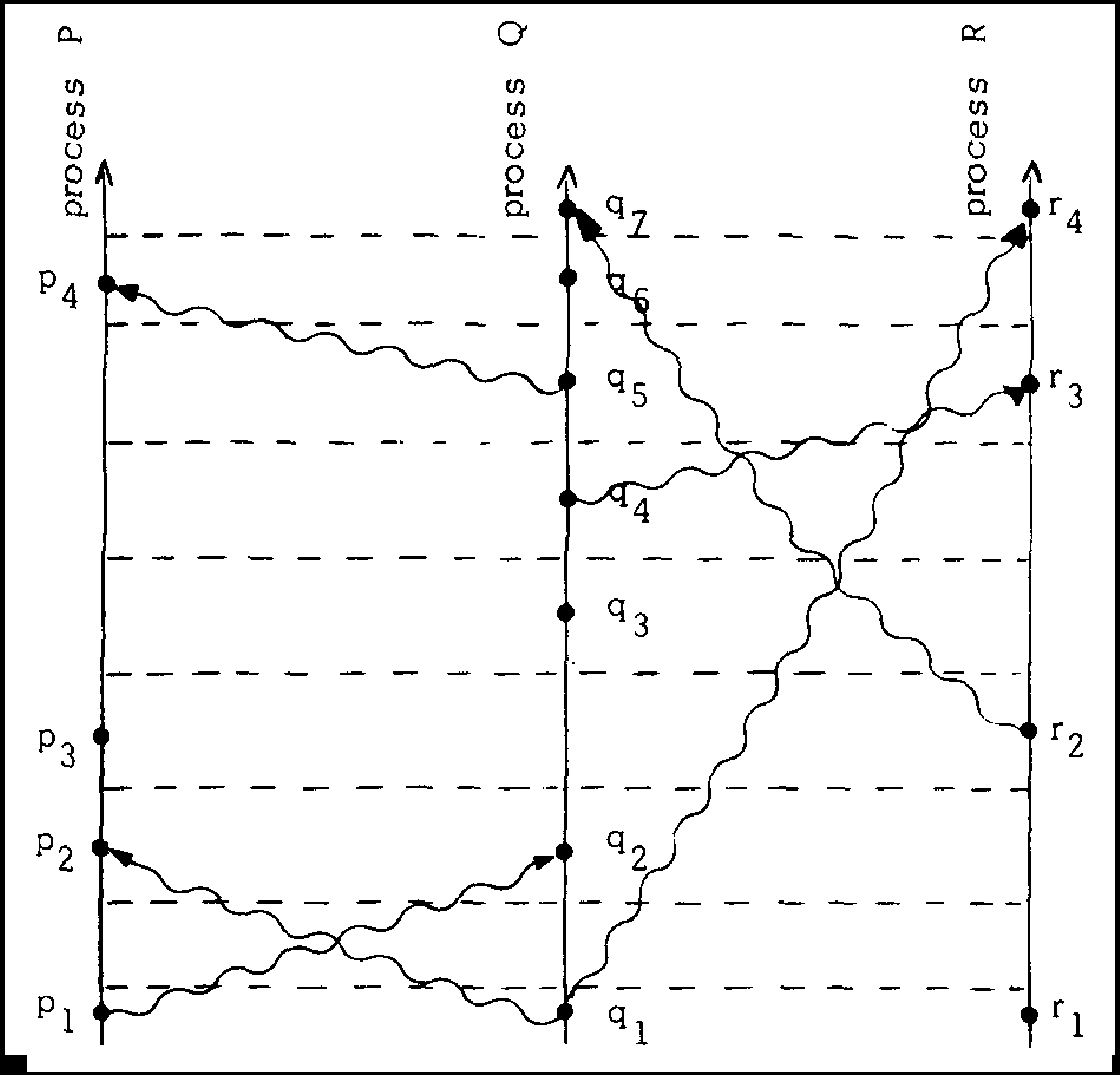
- Time goes up?
- Who ordered that?
- Time in quantum theory is a *parameter*.
  - First you do the experiment,
  - Then you look at you watch
- This is just as much nonsense as in the GR case where we impose a direction of the light cone from outside from our human perspective



DÆ

Time in Lamport Theory

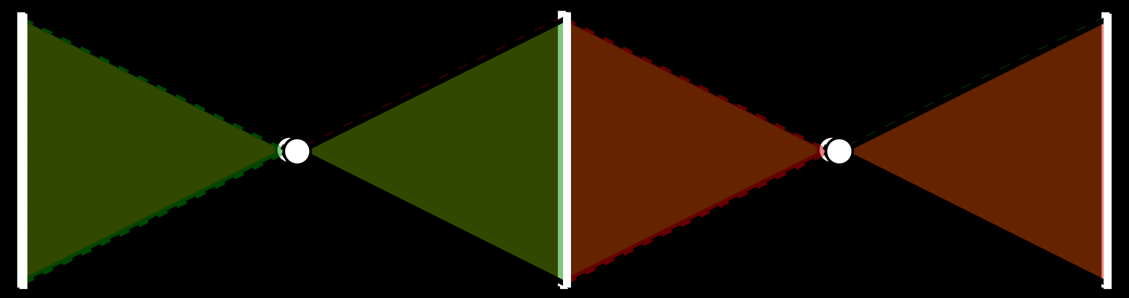
- Lamport's Logical Timestamps (Definite Causal Order (DCO))



Big causal effects

Alternating Causality

- Alice and Bob don't evolve against a background of Newtonian Time, or Minkowski time
- Human beings imposed their sense of past present and future on a background that doesn't exist



Fundamentals: Shannon Channel

1. Probability between 0 and 1. (Non-negative)
2. What does probability > 1 mean (See refs in binder)
3. What does negative probability mean?
4. What does complex probability mean?
5. How does Bayesian Probability enter into this ?

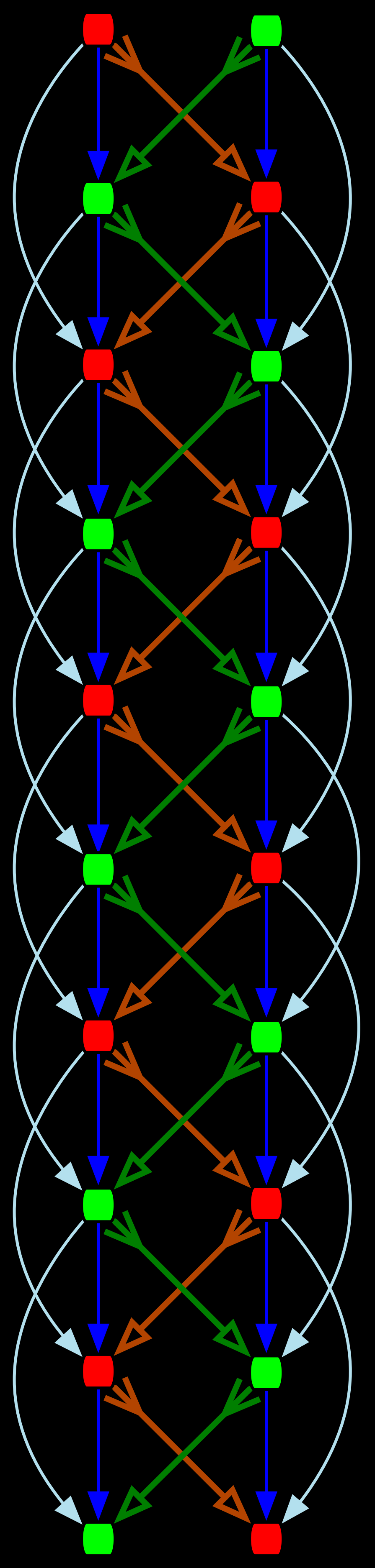
$$P(H \mid E) = \frac{P(E \mid H) \times P(H)}{(E)}$$

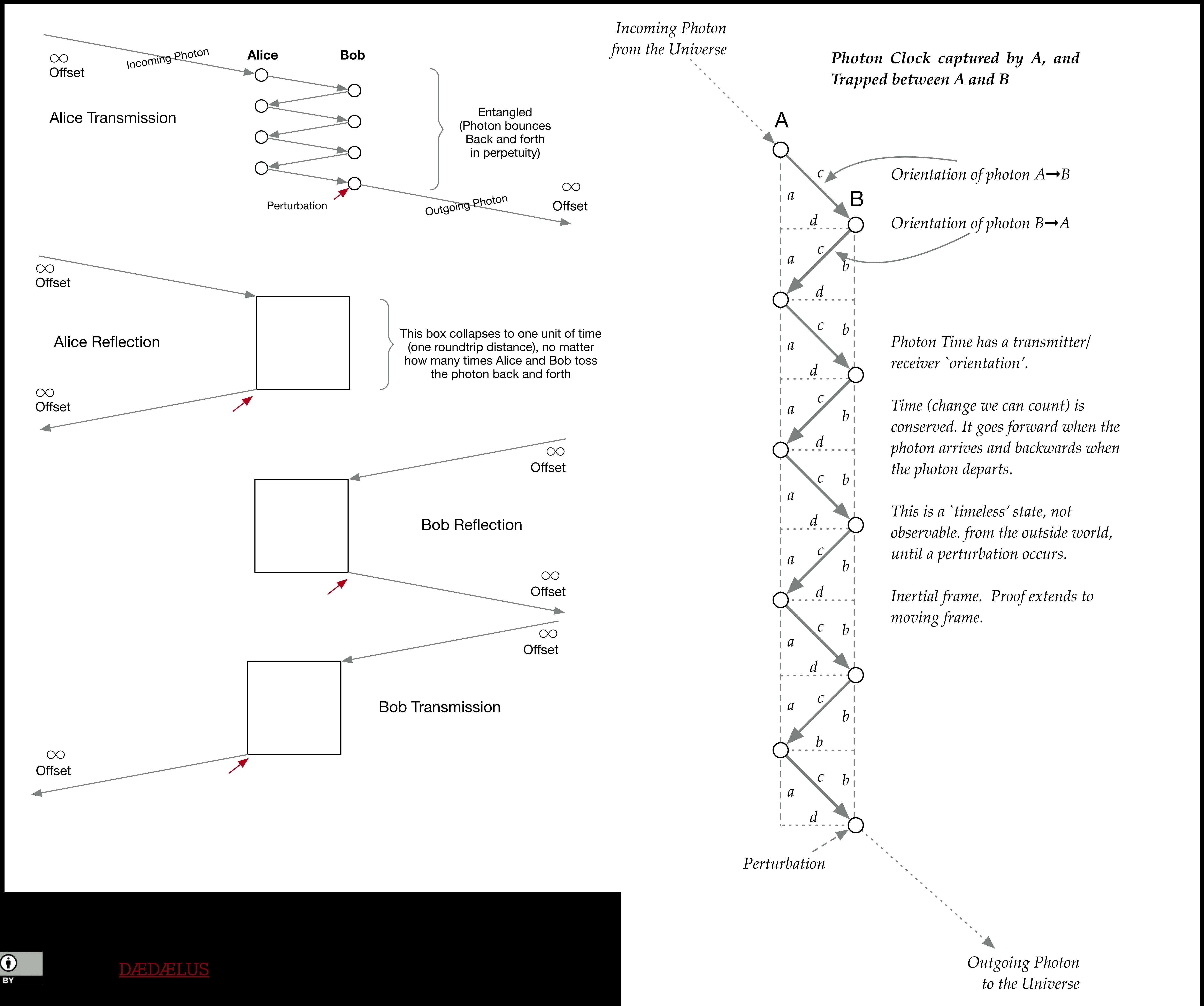
$$\uparrow$$

$$P(H) \times P(E \mid H) + P(-H) \times P(H \mid -H)$$

Causal Diamond

- Red and Green Alice and Bob





This box collapses to one unit of time (one roundtrip distance), no matter how many times Alice and Bob toss the photon back and forth

Photon Clock captured by A, and Trapped between A and B

Photon Time has a transmitter/receiver `orientation'.

Time (change we can count) is conserved. It goes forward when the photon arrives and backwards when the photon departs.

This is a `timeless' state, not observable. from the outside world, until a perturbation occurs.

Inertial frame. Proof extends to moving frame.

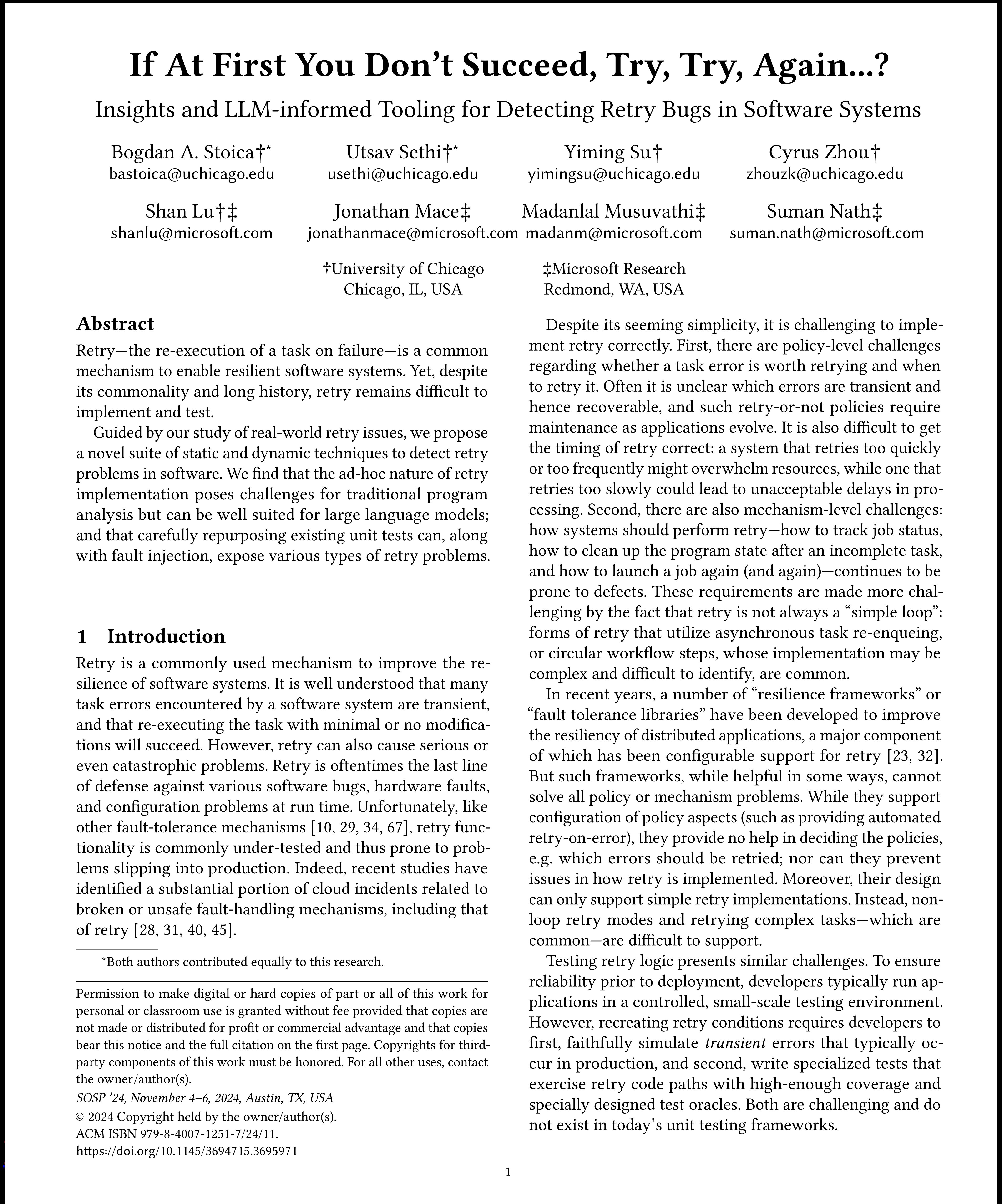
If at First you don’t succeed

SOSP, 2024

November 4-6, Austin Tx

“Retry—the re-execution of task on failure— is a common mechanism to enable resilient software systems. Yet, despite it’s commonality and long history, retry remains difficult to implement and test”

Timeouts and Retry is the root of all evil in distributed systems



If At First You Don’t Succeed, Try, Try, Again...?

Insights and LLM-informed Tooling for Detecting Retry Bugs in Software Systems

Bogdan A. Stoica†* bastoica@uchicago.edu
Utsav Sethi†* usethi@uchicago.edu
Yiming Su† yimingsu@uchicago.edu
Cyrus Zhou† zhouzk@uchicago.edu
Shan Lu†‡ shanlu@microsoft.com
Jonathan Mace‡ jonathanmace@microsoft.com
Madanlal Musuvathi‡ madanm@microsoft.com
Suman Nath‡ suman.nath@microsoft.com

†University of Chicago
Chicago, IL, USA

‡Microsoft Research
Redmond, WA, USA

Abstract

Retry—the re-execution of a task on failure—is a common mechanism to enable resilient software systems. Yet, despite its commonality and long history, retry remains difficult to implement and test.

Guided by our study of real-world retry issues, we propose a novel suite of static and dynamic techniques to detect retry problems in software. We find that the ad-hoc nature of retry implementation poses challenges for traditional program analysis but can be well suited for large language models; and that carefully repurposing existing unit tests can, along with fault injection, expose various types of retry problems.

1 Introduction

Retry is a commonly used mechanism to improve the resilience of software systems. It is well understood that many task errors encountered by a software system are transient, and that re-executing the task with minimal or no modifications will succeed. However, retry can also cause serious or even catastrophic problems. Retry is oftentimes the last line of defense against various software bugs, hardware faults, and configuration problems at run time. Unfortunately, like other fault-tolerance mechanisms [10, 29, 34, 67], retry functionality is commonly under-tested and thus prone to problems slipping into production. Indeed, recent studies have identified a substantial portion of cloud incidents related to broken or unsafe fault-handling mechanisms, including that of retry [28, 31, 40, 45].

Despite its seeming simplicity, it is challenging to implement retry correctly. First, there are policy-level challenges regarding whether a task error is worth retrying and when to retry it. Often it is unclear which errors are transient and hence recoverable, and such retry-or-not policies require maintenance as applications evolve. It is also difficult to get the timing of retry correct: a system that retries too quickly or too frequently might overwhelm resources, while one that retries too slowly could lead to unacceptable delays in processing. Second, there are also mechanism-level challenges: how systems should perform retry—how to track job status, how to clean up the program state after an incomplete task, and how to launch a job again (and again)—continues to be prone to defects. These requirements are made more challenging by the fact that retry is not always a “simple loop”: forms of retry that utilize asynchronous task re-enqueing, or circular workflow steps, whose implementation may be complex and difficult to identify, are common.

In recent years, a number of “resilience frameworks” or “fault tolerance libraries” have been developed to improve the resiliency of distributed applications, a major component of which has been configurable support for retry [23, 32]. But such frameworks, while helpful in some ways, cannot solve all policy or mechanism problems. While they support configuration of policy aspects (such as providing automated retry-on-error), they provide no help in deciding the policies, e.g. which errors should be retried; nor can they prevent issues in how retry is implemented. Moreover, their design can only support simple retry implementations. Instead, non-loop retry modes and retrying complex tasks—which are common—are difficult to support.

Testing retry logic presents similar challenges. To ensure reliability prior to deployment, developers typically run applications in a controlled, small-scale testing environment. However, recreating retry conditions requires developers to first, faithfully simulate *transient* errors that typically occur in production, and second, write specialized tests that exercise retry code paths with high-enough coverage and specially designed test oracles. Both are challenging and do not exist in today’s unit testing frameworks.

*Both authors contributed equally to this research.

SOSP ’24, November 4–6, 2024, Austin, TX, USA
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1251-7/24/11.
https://doi.org/10.1145/3694715.3695971

1

If at First you don't succeed

Retry bugs?

It's not a bug in the code, it's a bug in our assumptions

- False positives (timeout too soon), causing unnecessary smash and restart
- False negatives (causes slowdowns, limpware, metastable failures and corrupted data structures)

Orchestrating retries isn't going to save us

Randomizing retries isn't going to save us

Cancellation isn't going to save us, unless we can reversibly "retrieve" not yet captured information

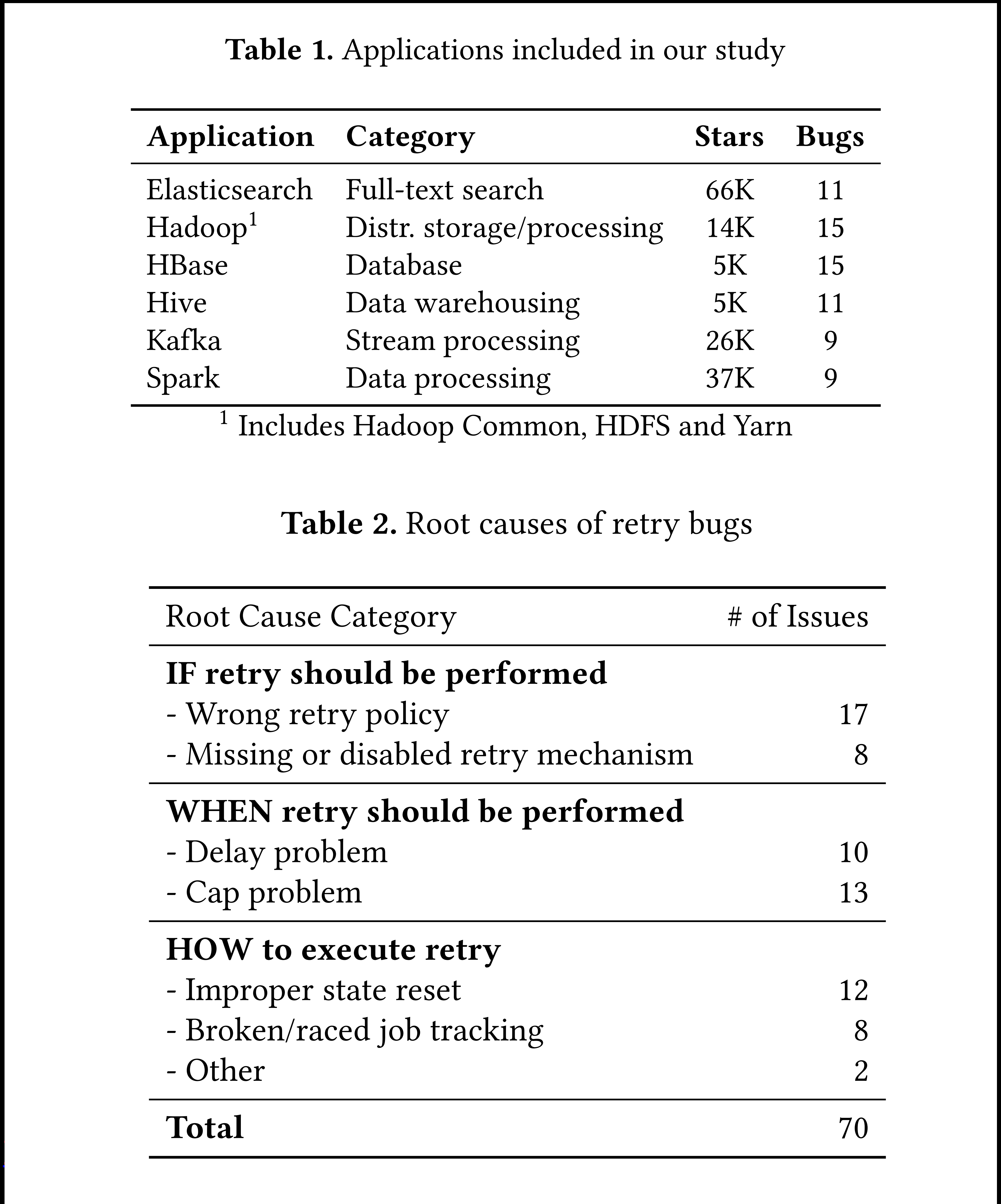


Table 1. Applications included in our study

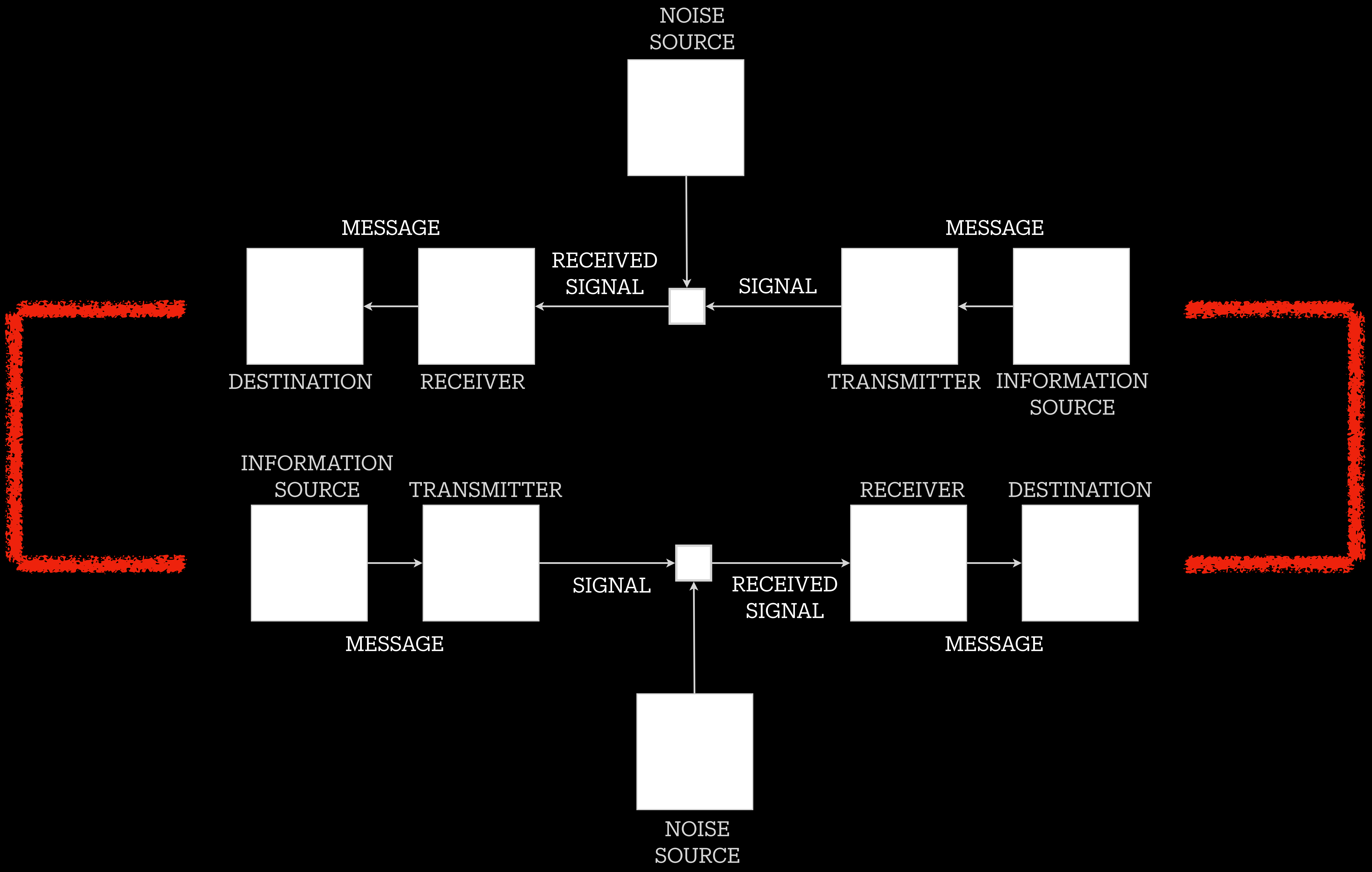
Application	**Category**	**Stars**	**Bugs**
Elasticsearch	Full-text search	66K	11
Hadoop[1]	Distr. storage/processing	14K	15
HBase	Database	5K	15
Hive	Data warehousing	5K	11
Kafka	Stream processing	26K	9
Spark	Data processing	37K	9

[1] Includes Hadoop Common, HDFS and Yarn

Table 2. Root causes of retry bugs

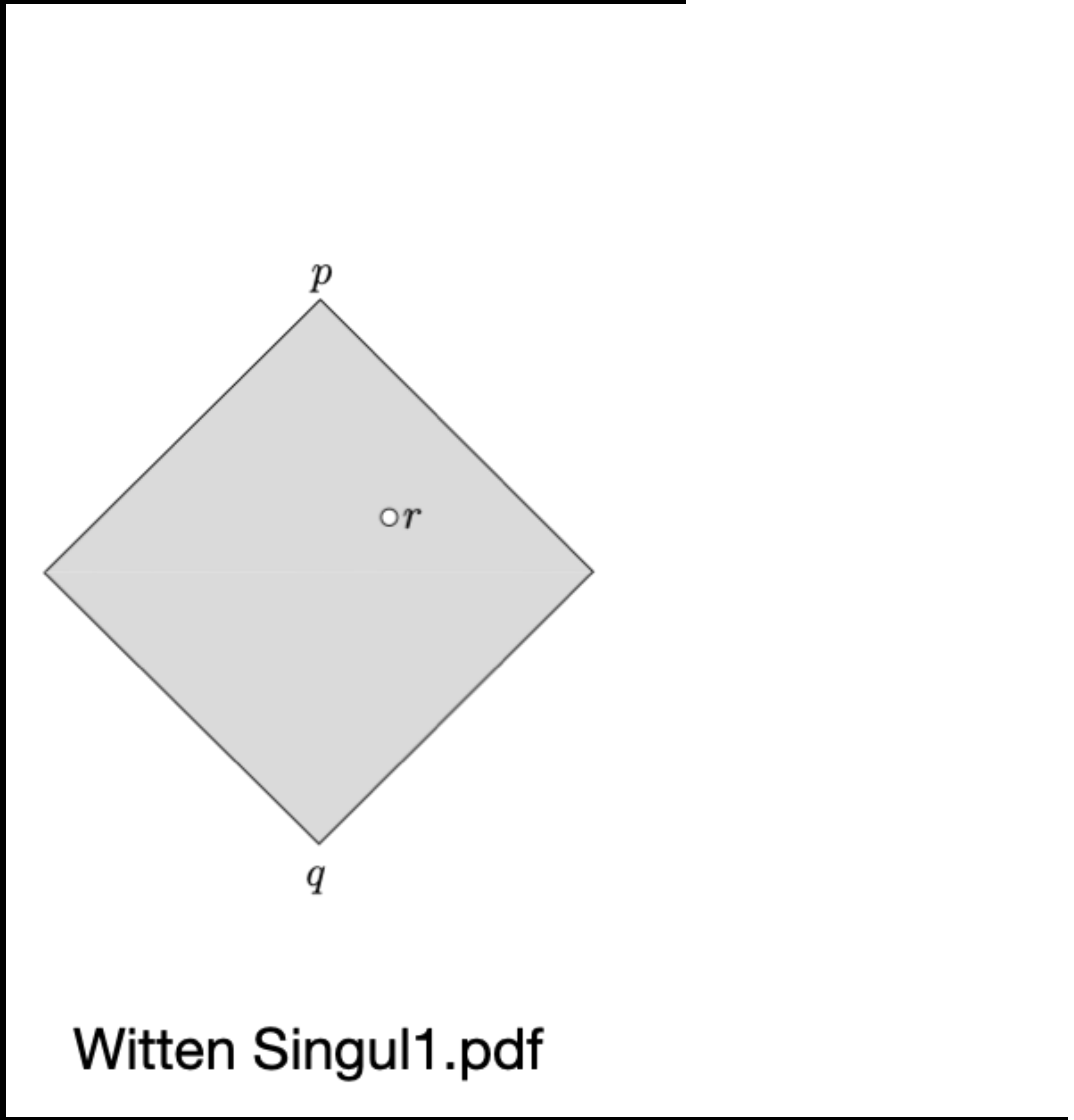
Root Cause Category	# of Issues
IF retry should be performed	
- Wrong retry policy	17
- Missing or disabled retry mechanism	8
WHEN retry should be performed	
- Delay problem	10
- Cap problem	13
HOW to execute retry	
- Improper state reset	12
- Broken/raced job tracking	8
- Other	2
Total	70

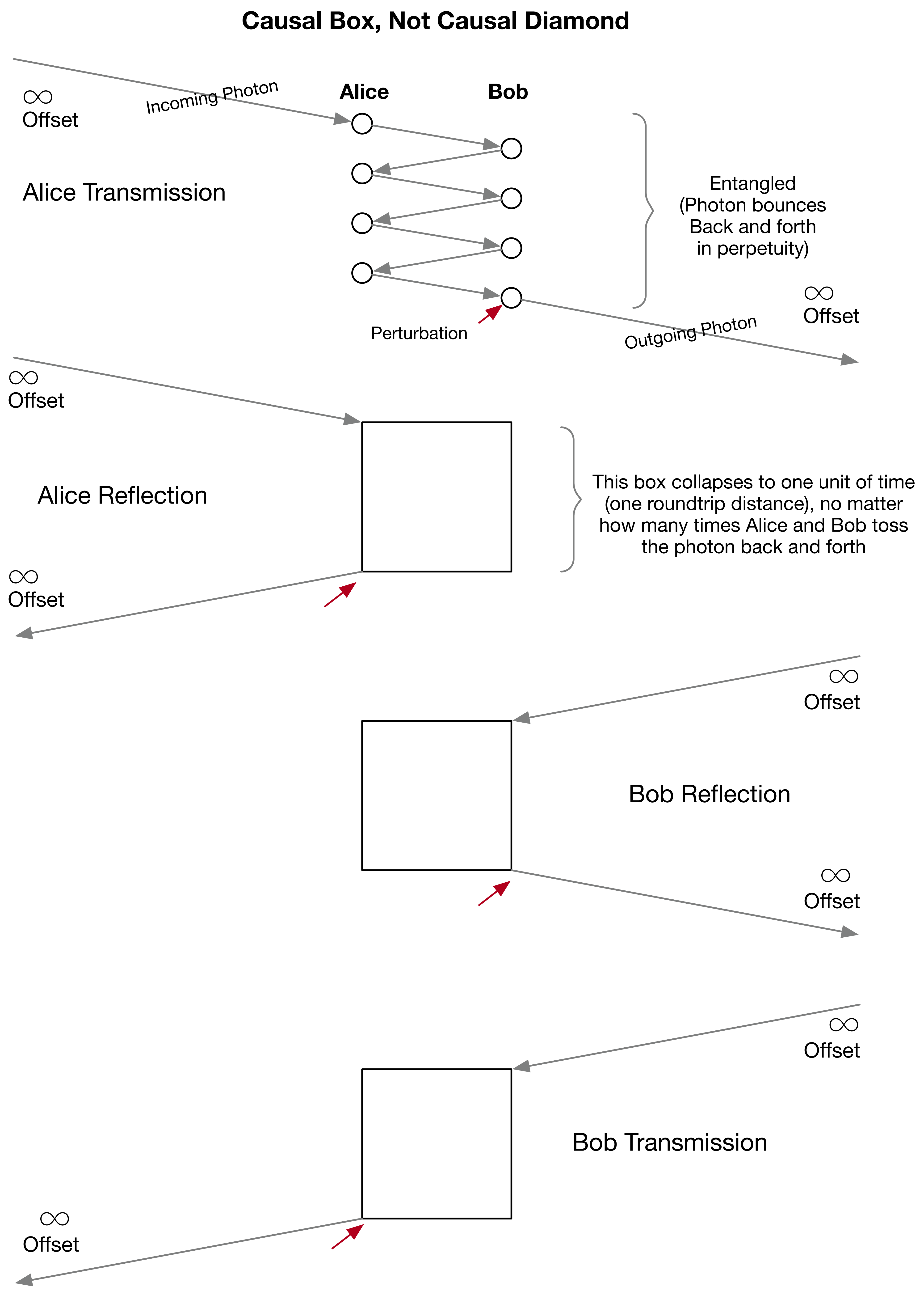
Back to Back Shannon Channels



Causal Box not Causal Diamond?

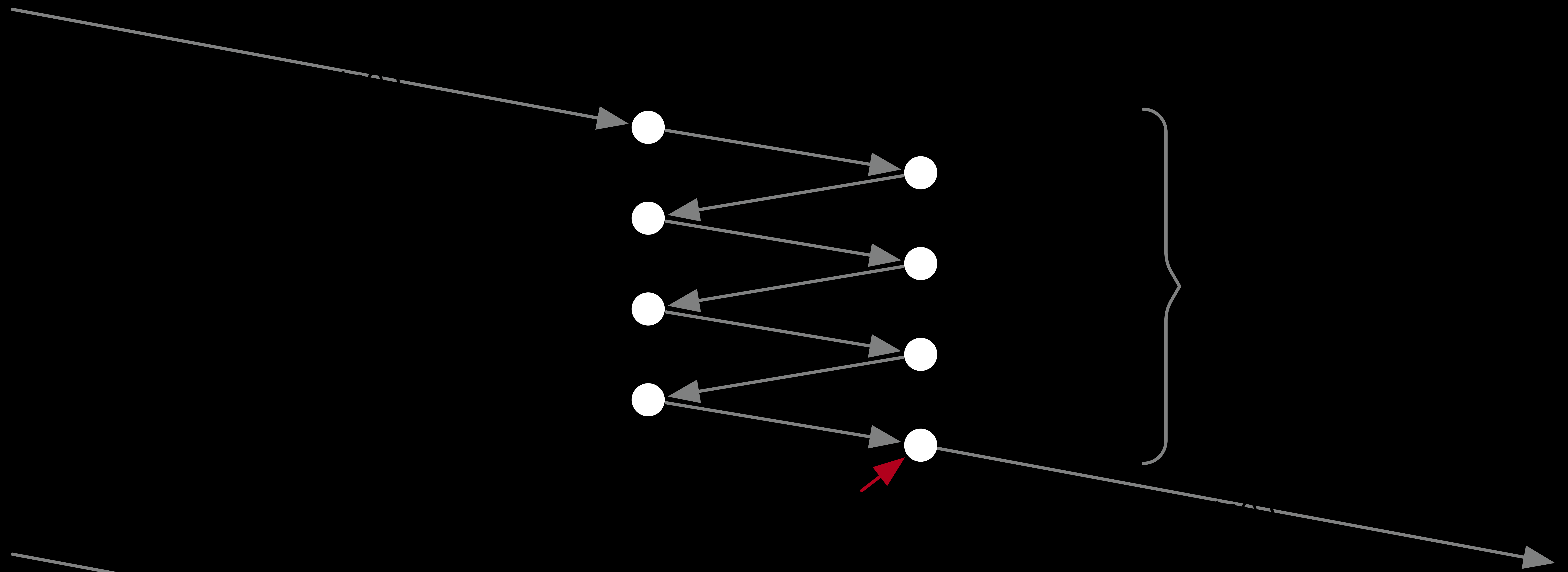
- Ed Witten: [Light Rays, Singularities and all That]

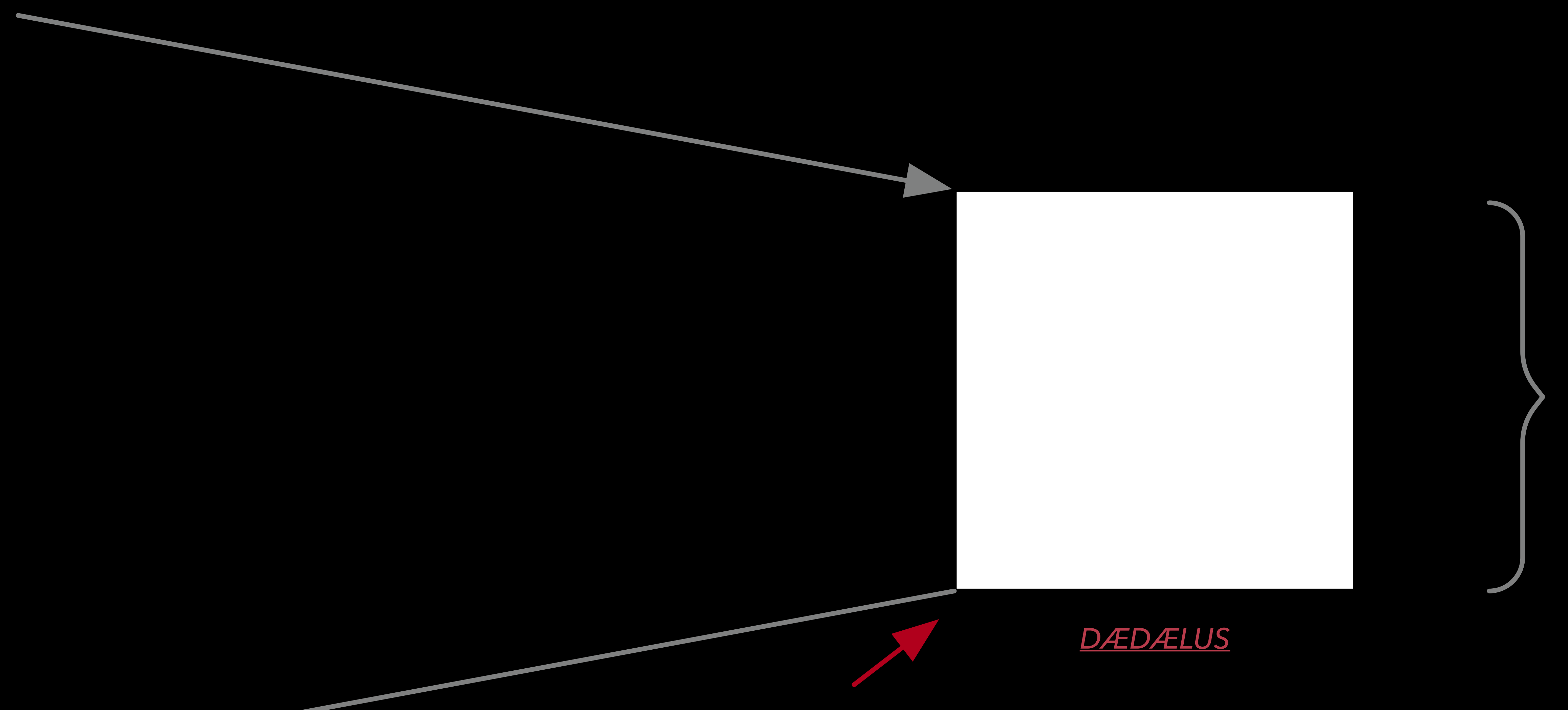



Causal Box, Not Causal Diamond
∞ Offset
Incoming Photon
Alice
Bob
Alice Transmission
Entangled (Photon bounces Back and forth in perpetuity)
∞ Offset
Perturbation
Outgoing Photon
∞ Offset
Alice Reflection
This box collapses to one unit of time (one roundtrip distance), no matter how many times Alice and Bob toss the photon back and forth
∞ Offset
∞ Offset
Bob Reflection
∞ Offset
∞ Offset
Bob Transmission
∞ Offset

DÆDÆLUS

Causal Diamond or Causal Box?

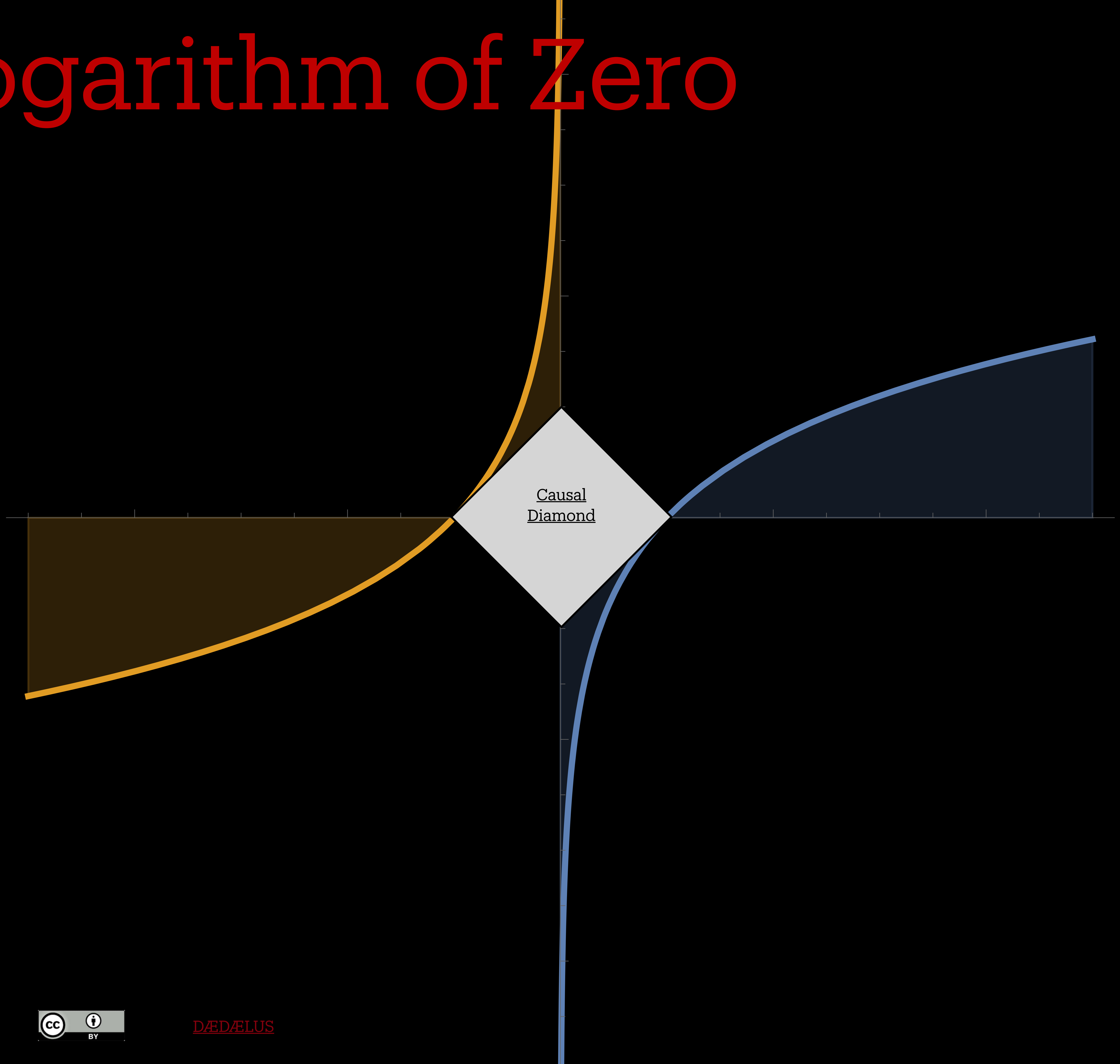




DÆDÆLUS

The logarithm of Zero

- No background of time
- The "orientation" of Alice and Bob is symmetric
- Are we "sending" information on a Shannon Channel, or are we receiving information on a return Shannon Channel?
- A causal diamond



Questions?

Timeout and Retry (TAR) is The Root of all Evil
In Distributed Systems

References

- Stanford 2016: “The Time-Less DataCenter" (November 2016)
  - Video: https://www.youtube.com/watch?v=IPTlTmH-YvQ
  - Slides: http://web.stanford.edu/class/ee380/Abstracts/161116-slides.pdf
  - Info: http://web.stanford.edu/class/ee380/Abstracts/161116.html
- Papers We Love 2016: “Lamport’s Unfinished Revolution” (July 2016 at GitHub HQ)
  - Video: https://www.youtube.com/watch?feature=youtu.be&t=32m27s&v=CWF3QnfihL4
  - Slides: https://speakerdeck.com/pborrill/time-clocks-and-the-reordering-of-events-pwl-san-francisco-14-jul-2016
  - Info: https://www.meetup.com/papers-we-love-too/events/228341271/
- Stanford 2014: Time in Physics, and Implications for Computer Science
  - Video: https://www.youtube.com/watch?v=SfvouFIVCmQ
  - Slides: http://web.stanford.edu/class/ee380/Abstracts/140416-Borrill-slides.pdf
  - Info: http://web.stanford.edu/class/ee380/Abstracts/091111.html
- A classical groupoid model for quantum networks
  - https://arxiv.org/abs/1707.00966