1. Theory

1.1 Introduction

It would be a mistake to assume conventional network concepts and terminology that you already know and love will remain unscathed in this project. We have no intention of reinventing the wheel, yet some new concepts and terminology will be necessary in order to escape the incrementalist momentum of the last five decades.

1. The word and the concept of TIME does not appear in this specification. This concept is the largest single source of misunderstanding in computer science today, so we eliminate that first. We replace this with intervals that are defined within sets (not on the real line). Definite Total Order (DTO), Definite Partial Order (DPO), Indefinite Partial Order (IPO). .
2. Claude Shannon described information as *surprisal*. We will call it Shanformation in this document to tear everyone away from old ideas that have been conflated for too long about "bits in memory or storage", hiding its deeper meaning of "the resolution of uncertainty".
3. We replace conventional notions of Error Detection and Correction (ECC, EDC, FEC, Parity, etc.) with a new concept, while not new, is widely misunderstood: Common Knowledge.
4. We replace conventional notions of liveness with a continuously circulating token, within which we define "logical simultaneity"
5. We are not shy of delving into Quantum Information Theory or Quantum Thermodynamics to find solutions to the problems in hardware and software infrastructure.

We carefully described in four presentations why the concept of time is widely misunderstood in the OCP TAP (Time Appliance Community). We respectfully request that you watch those presentations before insisting on timestamps or synchronized time in the context of Open Atomic Ethernet. Open Atomic Ethernet

Formal analysis and connections to the literature will appear in the index.

THERE IS NO GLOBAL DRUM BEAT - In Episodes 1 through 4 we expressed doubts about the common belief system of a Newtonian view of the world in this community. We showed how to think about race conditions, and why Timeouts and Retries (TAR) are the root of all evil. Our conclusion is that Timestamps are an Illusion. They can't be fixed by software. The quest for a single, consistent timeline across distributed systems collides with the reality that physics itself does not provide a universal notion of time—and in quantum mechanics (the machine code of our universe), there is no consistent causal order at all. We cannot, therefore, rely on this illusion of an irreversible drumbeat on an inaccessible "real line" to provide linear time order for events in our networked systems. Although timestamps will remain indispensable in engineering practice, we must recognize them as approximations rather than absolutes, and design our systems accordingly.

EPISODE 1 – There is no "now"

There is no now. You cannot synchronize clocks the way you think. Talk Originally given at the 2023 Asilomar Microcomputer Workshop presented live with Jonathan Gorard. Motivation: (1) To get people thinking about the nature of time and causality, as far removed from the Earth (and TAI/GPS) as possible. (2) To stimulate "First Principles Thinking" for Distributed Systems.

- Clocks can be disseminated, but require interaction to be synchronized:
- Simultaneity planes don't exist (except in an empty frozen universe) Einstein proved this over 100 years ago Why do we still think we can synchronize clocks?
- Network Time Protocol (NTP) and Precision Time Protocol (PTP) are causal TREES – choose your root, and how you do failover
- Entanglement and indefinite causal order are the new relativity (Not restricted to low relative velocities or atomic scales)
- We cannot assume spacetime is irreversible and monotonic
- Irreversibility and monotonicity is in the Eye of the Observer

EPISODE 2 - Hidden assumptions about causality lead to lost & corrupted data

When we think about clocks as an incrementing number, we are committing the FITO fallacy – "Forward In-Time-Only" Thinking - Counterfactuals, i.e., "events that could have occurred but eventually did not, play a unique role in quantum mechanics in that they exert causal effects despite their non-occurrence"

- Clock Synchronization Error is indistinguishable from Latency
- Irreversibility (Monotonicity) is an illusion not guaranteed by physics, unless we build Ancilla to explicitly manipulate causality
- Irreversibility and "causal order" are IN THE EYE OF THE OBSERVER—not guaranteed to be consistent across different observers

EPISODE 3 – Dynamic PTP hierarchy for causal failover

In Episode 1(What) & Episode 2 (Why) we showed how misunderstandings accumulate within a Newtonian framework of time, and how this leads to lost transactions and corrupted data. In this Episode we help the audience make the leap from Newtonian Time (what we know for certain that just ain't so) to Post-Newtonian Time (relativistic SR/GR, and QM — Indefinite Causal Order (ICO).

- PTP is widely available in Datacenters, we propose experiments to falsify beliefs about Newtonian Time.
- All is not lost. The excellent engineering behind PTP and PTM, can still be used with a different perspective, by using the clock hierarchy to build Causal Trees and reliable failover, to help address race conditions and achieve Exactly Once Semantics

EPISODE 4 – Why we can't have nice things in Distributed Systems

- 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 (no information) is minus infinity.
- Bayesian approaches require a prior belief, which can be unbounded (zero to infinity). 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 $p * log(p)$
- Measurements "appear" instantaneous because there is no background of time on which to measure anything. Timestamps don't help with causal order.

It *does* appear instantaneous to an observer, because when they receive a packet (or a photon in a detector), you capture information and turn it into knowledge (state in a register you can do something with)

1.1.1 Proof of Fallibility

> *"The alternating validation bit becoming the additionally the* alternation *bit for message transmission in one direction, while the alternation bit for the reverse directions serves additionally as a validation bit."* —SBW **?**

Lynch's scheme is constructed from independent simplex procedures

1.1.2 Framing the Vocabulary

In 1967 *Bartlett, Scantlebury & Wilkinson* (BSW) sketched the *alternating-bit protocol* (ABP): add one history bit to every frame, wait for an ACK that echoes it, and retransmit until the right ACK appears. ABP wraps an *unreliable* medium and presents a service that looks reliable—even *infallible* in steady state.

Nancy Lynch later formalised these ideas with the *I/O-automaton*: a fallible channel is one whose execution may deviate from the specification, subject only to fairness.

Term	Meaning
Unreliable	Frames may be lost.
Fallible	Channel may violate any promise (drop, duplicate, reorder, corrupt).
Reliable	No drops in steady state; recovery still required.
Infallible	No violations ever; no recovery logic needed above the link.

Figure 1.1: Taxonomy of link qualities.

1.1.3 Lessons from BSW and Lynch

- *Make reliability local*. ABP attaches one bit; InfiniBand embeds a few more. End-to-end recovery alone expands the failure scope.
- *Fail fast*. InfiniBand retransmits on explicit NACK within microseconds; Ethernet traditionally converts a microsecond drop into a millisecond TCP timeout.
- *Separate reliability from recovery*. Even a reliable link needs a failsafe plan; design that plan explicitly.

> BSW showed that a single alternating bit can tame a capricious wire; Lynch supplied the proof rules. InfiniBand adopted both insights in hardware and delivers a fabric whose normal behavior feels *infallible*. Classic Ethernet remains best-effort—unreliable and fallible—and relies on upper layers for recovery. Bridging that gap means absorbing more of the ABP playbook at the link: credits, link-level retransmission, and tight ACK/NACK loops that shrink recovery from milliseconds to microseconds.

1.1.4 IP and Patent Implications

The core concept of credit-based flow control is now public domain, but a handful of post-2005 implementation patents remain enforceable through at least 2036. Modern designs should audit those families but can build freely on the expired foundational work.

Practical Take-Aways

1. **Freedom to Operate.** A straightforward "one credit = one buffer" design can rely on the expired Intel/Compaq and Brocade patents for prior-art cover. Avoid features identical to the still-active patents or license them.
2. **Design Around.** Active claims tend to be narrow. You can sidestep the Mellanox "macro credit" idea by limiting link span or by using rate-based pacing instead of credit aggregation.

1.2 Time, Relativity, and Ethernet

By now many people are realizing that there's something amiss in our understanding of the nature of time in computer science. We are learning to think more carefully about whether, or how, the theories of Special Relativity (SR) and General Relativity (GR) might be involved. We are no longer willing to just sweep them under the rug with the assumption of an inertial frame, which is invalid a world where our computers reside: on a rotating sphere, orbiting a star, and in a meaningful gravitational field.

Strawman objections to SR assume we can ignore its effects in the low relative velocity of objects in our environment. Even computers in cars, airplanes and rocketships, where . But this perspective fails to appreciate the deeper implication that clocks cannot be synchronized in principle; and will therefore be problematic in practice.

Strawman objections to QM claim that it applies only at atomic scales. But the 2022 Nobel prize for Quantum Entanglement demonstrated the scientific community's confidence that Einstein's spooky action at a distance is real, measurable, and a foundational property of the universe we live in — and it's effects manifest at macroscopic scales.

I am either fooling myself (which is highly likely given how easy it is for any of us to fool ourselves), or I am among several (perhaps many) seeking both practical and deterministic mechanisms to manage reliable and consistent event ordering in computers.

Our motivation, as engineers, is to solve problems in distributed systems that undermine consistency, reliability, availability, and performance of databases and file synchronization in conventional cloud infrastructures, and show how it does not need to be the case that they inevitably lose or corrupt data.

The outcome we seek is a software infrastructure for distributed systems based on algorithms whose assumptions about causality go beyond Newtonian and Minkowski spacetime.

1.2.1 A New Perspective:

Photons arriving at the JWST have traversed the largest "distances" in the known universe, but they have done so in zero time: the proper time from a photon's perspective is always zero, no matter how far it has come.

One reason we may be confused about the nature of time is because the concept of velocity takes us down the wrong path:

- Distance is not fundamental,
- Time is not fundamental,
- Therefore why should distance divided by time or it's calculus equivalent be fundamental?

Perhaps this is because the speed of light (SOL) is the pivot around which space and time revolve.

1.2.2 Key problem: Minkowski Error

- Photons don't carry timestamps. There are no "Minkowski manifold coordinates" transmitted (emitted) alongside or with the photon.

- There are no Minkowski manifold coordinates at 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 create a background using NTP, PTP, White Rabbit or Sync E are doomed to fail. This style of thinking misleads us about the nature of causality in our algorithms and prevents us from building reliable distributed systems, i.e., those that don't silently lose or corrupt data structures—at least not without warning us explicitly that the atomicity assumptions in our subtransaction sequences were violated.

1.2.3 OWSOL

It is impossible to measure the One-Way Speed of light (OWSOL) [Lewis & Barnes]. Not only does the Einstein convention assume an average (1/2) of the total (roundtrip) time for what an emitter will measure, it also implies an impractical inertial frame, which exists only in an empty frozen universe — or a specifically engineered pair of mirrors in a rigid frame like Ligo.

Even with 'uniform' motion between one mirror and its partner in this measuring system, we must do more than relate the proper time of Alice to the proper time of Bob. But this now implies a Doppler shift, which requires some 'integration time' to resolve the uncertainty between the energy (wavelength) and whatever units of time the originator chooses to count as their 'measurement'.

Whenever the "duration" (time interval) between local clock ticks becomes comparable with the "duration" (space interval) between the mirrors, ambiguity will creep in because of a race condition between the arrival of the photon and the next tick of the local clock.

One way to overcome this race condition, that asynchronous circuits in all computer systems suffer from, is to think about clocks differently. Instead of each side having their own clock and attempting to measure the duration (interval) on a timeline independently of the other party, we use photons themselves as clocks, as Einstein implied. We achieve 'Temporal Intimacy' by having each arrival event trigger a reaction (sending event) in a circular, self re-generating relationship. We hesitate to call this a sequence because this interaction is completely symmetric, there is no notion of a direction of time.

Did Alice cause Bob or did Bob cause Alice? This 'alternating causality' is equivalent, from the perspective of each party, to time alternately going backwards and then forwards and then backwards again . . . in perpetuity.

This line of reasoning highlights the problem of trying to synchronize clocks in our datacenters. The OCP TAP project demonstrates considerable sophistication with experts in the engineering community trying to grapple with this problem.

Unfortunately, these attempts are doomed to failure, there are many ways to describe why, using established physics (SR & GR) that have many subtle and fascinating insights missing from the school-child rendition of overly simplified models of the physics.

There remains, particularly in SR, the thorny inconsistencies of a real universe that contains matter, radiation and dark energy.

1.2.4 Alternative Narrative on Anisotropy

When a photon leaves one "system" for the first time, velocity is not defined because both distance and time are not definable. Photon time is always zero, no matter how far the photon travels. Until a photon "arrives", distance does not exist. The first observer (receiver) of that photon is also unable to determine distance between the emitter and itself.

However, if that first receiver acts as a mirror and reflects the photon back to the original emitter, then it becomes possible (in principle) for the original emitter to determine the distance of two-way traversal, by referencing some local mechanism that can do some counting on its behalf. As with all quantum clocks, you need one clock in order to measure another.

This "lack of definability" of distance, in the one way traversal of a light ray, and the proper time of the photon being zero, provides an explanation of why we need a round trip to fully specify information transfer, and why one-way traversals are insufficient. This also provides an intuition for why we achieve correlations in Bell state measurements, and still require conventional SOL transfers for signaling. This corresponds to the Lewis and Barnes proof of the extreme case of anisotropy, where the SOL is infinite in one direction and $c/2$ in the other.

1.2.5 Indefinite, not infinite

Instead of a Newtonian (infinite SOL), the inbound photon from the first emitter, can we obtain an equivalent result by recognizing the initial one-way traversal as "indefinite"? By including Lewis and Barnes' bounds of both zero and infinite inf in the extreme anisotropic case:

$$C_{\pm} = \frac{1}{1 \mp \kappa}$$

where $\kappa = 1$ corresponds to anisotropic , and $\kappa = 0$ corresponds to an isotropic SOL.

This perspective on space time being "indefinite" appears consistent with recent measurements in the laboratory which decisively demonstrated (to almost 7 standard deviations) the indefinite arrival order of a quantum witness in the quantum switch configurations [See: Rubino et al: Experimental verification of an indefinite causal order]. Although this may be more about indefiniteness in the direction of time.

1.2.6 Time, Causality and Clocks

The ItsAboutTime.club podcasts (and the Mulligan Stew follow-on) are discussions with computer scientists, physicists, mathematicians and practicing engineers on how to transcend our human limited view of a single timeline going from the infinite past to the infinite future. We explore how to replace Newtonian time with a new intellectual vantage point based on "Kausality" (we hesitate to use the word Causality, because we don't want it to be obfuscated by the many confused definitions in the computer science literature).

1.2.7 Time is change we can count

The modern rendition of Aristotle's definition of time is expressed in Quantum information Theory as mutual information: But wait! Wasn't mutual information already in Claude Shannon's Mathematical Theory of Communication?" Well, yes, it was!

1.2.8 What have we forgotten since Shannon?

- Information is not binary zeros and ones, in registers, in memory or on disk.
- Although this may be closer, information is also not bits on the wire, launched by Ethernet transmitters, to get lost and never be seen again.
- Information is surprisal — something we didn't expect. As Niel Gershenfeld taught us in his book - if I tell you the Sun is going to rise tomorrow morning, that would not be information. If I told

you the Sun was not going to rise tomorrow morning, that would be information. Information is the answer to a yes/no question. This is why bits (0's and 1's) are an appropriate representation of this unexpectedness. Shannons, the unit of Information, represent a 50

- What we need most from the logic in our Ethernet links is the resolution of uncertainty. Did the transaction commit or abort? And more importantly, did it get stuck in some state of limbo, where data structures on one side or the other (or both) are inconsistent, and should not be exposed to the host processor on the other side of the CXL bus.

1.2.9 Clock Synchronization

We have come a long way from our original discussion about photons and the nature of time. We've learned that Minkowski Manifolds don't exist, except in our imagination, and that a 'background' timeline of timestamps, in NTP, PTP, White Rabbit or whatever, are also a figment of our imagination. Now let's put two and two together and see what that comes up to when we think about causality:

- One entity is destined to be alone in perpetuity. Without 'interactions' with others it will spend eternity being alone without ever comprehending the concept of loneliness.
- It takes two to Tango. Two entities are the minimum necessary to form an interaction, for "mutual" information to have any meaning. But although entanglement is monogamous, it also cannot display any dynamic behavior. Once each partner has found each other in the universe, they will remain entangled for the rest of eternity, and be only slightly better off than our single lonely entity.
- It takes three to Party. As Carlo Rovelli points out in Relative Quantum Mechanics, it takes a 3rd party (Charlie) to observe dynamical behavior, providing "all of the information possessed by a certain observer is stored in physical variables of that observer and thus is accessible by measurement to other observers".

1.2.10 Context

After thinking about the nature of time in the design of computer systems: processors, cache hierarchies, interconnects and networks for most of my career, I have found a repeating pattern: Computer scientists, are trained to think sequentially. We are in love with machines that do one damn thing after another: sequentially, monotonically, irreversibility, and their unfortunate pipe-dream: idempotency.

Distributed systems developers are accustomed to those illusions when building and scaling their systems, to be tolerant to all manner of failures as they grow, and exposed to unexpected traffic patterns.

1.2.11 Where is this Going?

Networks do exactly the wrong thing. They take a half duplex notion of "broadcasting" information blindly without "knowing" if the information is being received (and remembered) in receivers (observers). Before we get into the also confused concept of knowledge and common knowledge let's see what we get without interactions (the two-way exchange of Shannons).

- Nothing. Fire and forget is pointless, it can never be transactional. It takes two to tango and three to party.
- Half-duplex also breaks when we expect the other party to respond within our current context (the question we are asking) with a yes/no answer.
- Human interactions over two-way walkie-talkies, exhibit the same behavior. If the other party has gone to lunch or is taking a vacation, they may not even be listening.
- Their radio may not un-squelch because the transmit signal was too weak, or the receiver experienced "noise" which violated the signal to noise ratio that the receiver depends upon.
- The batteries may have run out. The receiver had lost "Temporal Intimacy" with the transmitter (emitter).
- This loss of "Temporal Intimacy" is also a deliberately engineered manifestation of optimizing for the wrong metric in system design, such as CPU cycles or network bandwidth.

  What Pat Helland refers to as Rip Van Winkle.

1.2.12 Examples Include:

- Garbage collectors going AWOL
- Virtual machines being suspended
- Containers' involuntary smash & restart when Kubernetes agents miss a heartbeat or gets confused about what question they were asking
- Processes being involuntarily pre-empted by the scheduler in the kernel
- Cache misses
- Page faults

The list goes on. We have difficulty calculating an "Interval" of time because even local notions of time on a single system rely on (what Edward Lee refers to as) "different timelines" and clocks within their architecture.

We now have processors with over 1,000 cores (Ampere ARM). Why do we insist on interrupting all of them to make them 'context switch' to optimize processor utilization? It's time our OS architecture moved on and recognize that causality and latency are the scarce resources, not CPU cycles or network bandwidth.

1.2.13 Where Else is this Going?

The biggest culprit in this tragedy of loss of Temporal Intimacy is not the processor, cache hierarchy or OS in a single node, it's the Network. IP (and TCP) try their best to cope with what they can get on top of lossy Ethernet in a conventional Clos network.

The End to End (E2E) principle works only for a simple packet sequence over an unreliable network. It was invented when the primary use case was file transfer (a finite stream of bytes sent to a printer).

It works abysmally poorly for transactions that require an unbroken sequence of atomic interactions to deterministically transfer Shannon Information from one node to another.

Note that we chose the word "transfer" here, not "copy". Our premise is that we cannot achieve exactly-once semantics without some mechanism to ensure conserved quantities and we cannot conserve quantities when we copy information willy-nilly, because that would make it impossible to keep track of Shannons, and account for them.

1.2.14 Ethernet Spacetime

Now that we've separately described Shannon information on an Ethernet link as surprisal, and photon traversals in spacetime as indeterminate in measurements of the one way speed of light (OWSOL), we now put the two together in Ethernet Spacetime.

The Alternating Causality (AC) protocol is already a staple of the networking industry, implemented originally as the Alternating Bit Protocol (ABP) or "Stop and Wait" (SAW) protocol, which preceded credit based flow control, commonplace in Infiniband, and now Ethernet.

Credit-based flow control has the semantics of conserved quantities (CQ), and one might consider building on this to try and achieve exactly-once semantics (EOS), but alas, current versions of Ethernet and Infiniband still rely on timeouts and retries to recover when liveness (Temporal Intimacy) is lost.

This happens because designers of switches and routers focus on delivering bandwidth.

Researchers tried to overcome the imposition-style behavior of protocols at the IP layer with promise-style behaviors in transport protocols like Homa. While this may improve the emergent behaviors of unexpected congestion events in the network, it is far from perfect and illuminates why congestion control needs to be addressed in L2 and not L3.

The Ultra Ethernet Consortium (UEC) plan to address the problem of congestion and dropped packets by more aggressively sending packets through the switched network via multiple paths. The extreme case being sending packets on all ports and hoping the receiver can sort out the mess at the other end using some form of reordering buffer. They also promote optimizations to more selectively fill holes in the reordering buffer instead of the go-back-N approach of TCP and Infiniband.

1.2.15 Bang Bang Bang Networking is inconsistent with the needs of distributed systems

Unfortunately the bang bang bang style of networking, resting ultimately on timeouts and retries, results in cascade failures — which manifest as Limpware, Metastable failures, and Grey failure.

This is in addition to, and exacerbates, the silent loss and corruption of data that we see everywhere in open-source databases and consensus tools to coordinate distributed systems.

Gossip protocols are the ultimate hammer in bang bang bang networking. Statistically maintaining liveness by having each node transitively connect its heartbeats to a graph of flat network partners to probabilistically improve temporal awareness, without ultimately achieving temporal intimacy. Gossip protocols are employed extensively at Amazon and in software like Hashicorp's Consul and Terraform. Unfortunately, gossip protocols also impose a nondeterministic amplification of bandwidth by sending redundant packets through the network. This further exacerbates the congestion, packet loss, which leads to more silent data loss and corruption.

1.2.16 Possible Solution?

In contradistinction to the industry efforts to manage congestion and packet loss, the Daedaelus team has conceived, designed and tested an actual Temporal Intimacy Protocol on Ethernet.

The breakthrough is eliminating timeouts and retries; by using an event-driven protocol where events are accounted for using the principle of Conserved Quantities (CQ) in each link. Accounting for token insertion and retirement is easy, when the application controls the spanning trees, instead of waiting for innovation to occur in the switches and routers, under the control of network operators.

Our solution is targeted on the Edge.

We see the potential for a much needed paradigm shift as we enter the Chiplet Revolution and learn to rethink our notions of time and event order in distributed systems must be deterministic on demand.

1.2.17 Temporal Intimacy is not Simultaneity

We know that the Conventionality of Simultaneity must be replaced by something not dependent on Newtonian time, or Minkowski Spacetime. According to John Norton:

> "The conventionality of simultaneity pertains to judgments of simultaneity of distant events in just one inertial frame. In asserts that there is no single, correct judgment of simultaneity. Rather, in each inertial frame, we have broad freedom in assigning simultaneity to pairs of events. In the same frame of reference, one person may assign relations of simultaneity one way; another person may do it differently. Within some limits, neither is factually wrong, according to the conventionality thesis, for there is no unique fact of simultaneity in the world".

Temporal Intimacy requires two-way interactions, not one way (fire and forget) broadcasts. Otherwise, how does the emitter know if the absorber received the information?

1.2.18 Tentative Conclusions

Monotonicity is in the eye of the observer. Clock Synchronization Error is indistinguishable from Latency. A femto second interval is an eternity on the real line.

1.3 Two-State Vector Formalism

The *Two-State Vector Formalism* (TSVF), developed by Aharonov and collaborators, describes a quantum system using both a forward-evolving state vector from the past and a backward-evolving vector from the future, forming a complete description of the system between two time boundaries.

This formalism maps intriguingly well onto the two-phase behavior of supervised learning:

- **Forward propagation:** evolving the input forward through the network to predict an output.
- **Backpropagation:** retroactively applying a loss function at the output and propagating error information backward through the network to adjust weights.

Forward propagation plays the role of the forward-evolving quantum state $|\psi(t)\rangle$, and backpropagation corresponds to the backward-evolving dual state $\langle\phi(t)|$. Together, they constrain the learning dynamics at each layer via a two-state viewpoint.

1.3.1 Forward Propagation as Forward Evolution

In TSVF:

$$|\psi(t)\rangle = U(t, t_0)|\psi(t_0)\rangle,$$

where U is the unitary time evolution operator from an initial preparation at t_0.

In machine learning:

$$a^{(l+1)} = f^{(l)}(W^{(l)}a^{(l)} + b^{(l)}),$$

where the activations $a^{(l)}$ propagate the input forward.

1.3.2 Backpropagation as Backward Evolution

In TSVF, a post-selected state $\langle\phi(t_1)|$ evolves backward:

$$\langle\phi(t)| = \langle\phi(t_1)|U(t_1, t),$$

complementing the forward-evolving $|\psi(t)\rangle$.

In neural nets:

$$\delta^{(l)} = (W^{(l+1)})^\top \delta^{(l+1)} \circ f'^{(l)}(z^{(l)}),$$

where $\delta^{(l)}$ encodes error at layer l and propagates backward to adjust weights.

1.3.3 Two-State Update Rule

In TSVF, expectation values take the form:

$$\langle A\rangle_w = \frac{\langle\phi|A|\psi\rangle}{\langle\phi|\psi\rangle},$$

and represent weak values or amplitudes constrained by both past and future states.

In machine learning, the update rule:

$$\Delta W^{(l)} \propto \delta^{(l)}(a^{(l-1)})^\top$$

depends on the forward activations $a^{(l-1)}$ and backward error signal $\delta^{(l)}$. Together, they act like a sandwich operator:

$$\text{Update}^{(l)} \sim \langle\phi^{(l)}|\text{operator}|\psi^{(l)}\rangle.$$

1.3.4 Time-Symmetric Learning View

Rather than treating backpropagation as a mere computational trick, TSVF offers a time-symmetric interpretation:

- Both the input and the desired output state determine the intermediate learning dynamics.
- Each layer mediates between past input and future supervision, forming a time-bridging node.

Concept	Neural Network	TSVF QM
Initial input	x	$\lvert\psi(t_0)\rangle$
Prediction process	Forward prop	$U(t, t_0)$ evolution
Target supervision	Loss function	Post-selection $\langle\phi(t_1)\rvert$
Error signal	$\delta^{(l)}$	$\langle\phi(t)\rvert$
Intermediate activity	$a^{(l)}$	$\lvert\psi(t)\rangle$
Weight update	$\delta^{(l)}(a^{(l-1)})^\top$	$\langle\phi\lvert A\rvert\psi\rangle$

Table 1.1: Analogies between supervised learning and the Two-State Vector Formalism (TSVF) in quantum mechanics.

1.3.5 FITO Thinking vs. Time Symmetry

Most machine learning frameworks assume *Forward-In-Time-Only (FITO)* causality: input causes output, and learning proceeds only by adjusting from past to future. TSVF suggests a richer model:

- Supervision from the future constrains the learning of the past.
- This bidirectional model aligns with concepts from goal-driven behavior and active inference.

1.3.6 Conclusion

The TSVF reframing of forward and backpropagation illuminates the deeper time-symmetric structure underlying learning. Far from being just a computational trick, backpropagation can be seen as a physical dual to forward propagation—both necessary to fully specify a learning system between two boundary conditions.

This is highly relevant to our model of the protocol, where we use reversible state machines to specify forward propagation, with whatever reversible glitches might be needed to handle failures are implemented as reversible steps, and the backpropagation as the credit-based flow control mechanism.

Because they are completely symmetric, packets being sent and packets being unsent are fully managed by the flow control system.

1.4 TIKTYKTIK

From: ./AE-Specifications-ETH/standalone/TIKTYKTIK.tex

TIKTYKTIK is like the alternating-bit and stop-and-wait protocols in that receipt of a packet over a link is acknowledged over that link with a “signal” packet. In that sense, these three protocols implement credit based flow control, which simplifies buffer management and makes it possible to not have to drop packets when there is a lot of traffic.

TIKTYKTIK adds a second round trip, which provides partial common knowledge helpful for recovery from link failures. This document walks through TIKTYKTIK showing how that common knowledge is used. First look at the various stages of common knowledge as the protocol runs without failure when Alice sends a packet to Bob.

1. **Alice sends the packet to Bob**
   - Alice doesn't know if Bob received the packet
   - Bob does not know the packet exists
2. **Bob receives the packet**
   - Bob knows that Alice doesn't know that Bob received the packet
3. **Bob sends a signal to Alice**
   - Bob doesn't know if Alice knows that Bob received the packet
4. **Alice receives the signal**
   - Alice knows that Bob received the packet
   - Alice knows that Bob doesn't know that Alice knows that Bob received the packet
5. **Alice sends the signal**
   - Alice doesn't know if Bob knows that Alice knows that Bob received the packet
6. **Bob receives the signal**
   - Bob knows that Alice knows that Bob received the packet
   - Bob doesn't know if Alice knows that Bob knows that Alice knows that Bob received the packet
   - **Bob can forward the packet**
7. **Alice receives the signal**
   - Alice knows that Bob knows that Alice knows that Bob received the packet
   - **Alice can delete her copy of the packet**

This common knowledge is not needed if links never fail. Alice could delete the packet as soon as she sent it, and Bob could forward it as soon as he received it. That's what current systems do and why it's so hard to recover from a link failure.

A data packet can serve as a signal.[1] Links can fail in a number of ways. If they physically break or are unplugged, the PHY detects the lost of electrical signal and informs the higher layers. Links can also fail silently, such as when the NIC misbehaves. They can also fail in one direction but not the other. Silent failures can be detected in these protocols because a signal will never be received in either direction. In that sense, there is a level of common knowledge on a link failure. In what follows, I'll describe what happens when Alice wants to send a packet to Bob, but the link fails at various steps of the protocol. The link is no longer used once one of these failures occurs. (The link can

[1] A data packet can serve as a signal.

be used later after re-initializing the connection.)

1. **Alice has a packet to send when link fails**
   Alice: • Knows Bob doesn't have the packet
   • Knows Alice is responsible for it
   Bob: • Doesn't know the packet exists
2. **Alice sends the packet to Bob then link fails**
   Alice: • Doesn't know if Bob got the packet
   • Knows that Bob knows Alice is responsible
   Bob: • Doesn't know that the packet exists
3. **Bob receives the packet then link fails**
   Bob: • Knows that Alice knows that Alice is responsible
   Alice: • Alice doesn't know if Alice or Bob is responsible
4. **Bob sends signal then link fails**
   Bob: • Doesn't know if Alice got the signal
   • Doesn't know if Bob or Alice is responsible
   Alice: • Alice doesn't know if Alice or Bob is responsible
5. **Alice receives the signal then link fails**
   Alice: • Knows that Bob is responsible
   • Knows that Bob doesn't know he is responsible
   Bob: • Doesn't know if Alice got the signal
   • Doesn't know if Alice knows that Bob is responsible
6. **Alice sends signal then link fails**
   Alice: • Knows that Bob is responsible
   • Doesn't know if Bob knows he is responsible
   Bob: • Doesn't know if he's responsible for the packet
7. **Bob receives the signal then link fails**
   Bob: • Knows that Alice knows that Bob is responsible
   Alice: • Doesn't know if Bob knows that Bob is responsible
8. **Bob sends signal then link fails**
   Bob: • Knows that he is responsible
   • Doesn't know if Alice knows he is responsible
   Alice: • Knows that Bob is responsible
   • Doesn't know if Bob knows that Bob is responsible
9. **Alice receives signal**
   Alice: • Knows that Bob knows that he is responsible

At this point, Alice knows that Bob knows that Bob is responsible, and Bob knows that Alice knows that Bob is responsible.

Note: The items in red indicate where incomplete common knowledge needs to be resolved over the new path between Alice and Bob.

1.4.1 TIKTYKTIK: Feedback from ChatGPT

The analysis you provided explores how "common knowledge" operates within the TIKTYKTIK protocol, especially in the context of link failure recovery. The progression of common knowledge is explained through several stages, starting with Alice sending a packet to Bob and progressing through various acknowledgment signals and link failures.

Key observations:

1. Common Knowledge Phases: The stages you outlined show how knowledge evolves between Alice and Bob as they interact, especially when there's uncertainty about packet receipt and acknowledgment. These stages are useful for understanding how communication protocols can handle and resolve ambiguities.
2. Link Failures: You go into detail about different link failure scenarios, from Alice sending the packet before failure to Bob acknowledging receipt of the packet after a failure. Each failure step indicates the differing levels of knowledge held by Alice and Bob. These steps, especially when the link fails, highlight the challenges of network fault tolerance, and the importance of "common knowledge" for recovering from these failures.
3. Protocol Behavior with Failure: The distinction between knowledge of responsibility and knowledge of receipt is well captured. The incomplete common knowledge as the link fails suggests that TIKTYKTIK, by adding a second round-trip acknowledgment, resolves uncertainties that could otherwise hinder recovery.

Suggested Refinements:

Clarity: Consider rephrasing the stages of knowledge with more emphasis on the protocol mechanics and timing of messages. It could be helpful to lay out the flow of the protocol in a diagram or a clearer timeline to illustrate the evolution of knowledge.

Link Failures and Recovery: While the failure scenarios are well outlined, you could further enhance the analysis by providing examples of real-world failure cases and how TIKTYKTIK compares to other protocols in handling these failures. For example, you could contrast how TCP handles packet loss versus how TIKTYKTIK's multi-step acknowledgment approach helps resolve knowledge gaps.

Mathematical Notation: If possible, you could introduce formal mathematical notations to express the knowledge states. This would help readers better grasp the protocol's behavior in a more structured form.

1.5 Common Knowledge

In what follows, assume that node Alice is sending a packet to node Bob over the single, fallible link between them.

The Stop and Wait and alternating bit protocols provide credit based flow control using a single round trip. Bob is free to forward the packet as soon as it arrives, but Alice must wait for a signal from Bob before sending another packet. If the link breaks before Alice gets a signal from Bob, then Alice may forward the packet again, perhaps on another path. This behavior makes exactly-once, in-order delivery difficult to implement.

TIKTYKTYK is one round trip beyond stop and wait, which provides partial common knowledge that aids in recovery from failures. Bob cannot forward the packet until he receives the signal from Alice that completes the second round trip. A key point is that there are many times when both sides know that both sides know which of them is responsible for forwarding the packet. In the other cases, the partial common knowledge simplifies recovery. Alice and Bob use their partial common knowledge to ensure that any packet is only forwarded once, which is a key condition for exactly-once, in-order delivery.

There is minimal loss in latency, because Bob doesn't have to wait for the entire packet to arrive before signaling to Alce that the packet is arriving. He can do any integrity checks (CRC) while waiting for Alice's signal. Any loss in latency is compensated for by needing smaller buffers. There is minimal loss in bandwidth because the signal can be a data packet going in the other direction.

Sometimes links fail silently, which means a signal might not arrive. In those cases, he nodes will need some heuristic to decide when to stop waiting and declare the link dead. Fortunately, this heuristic can be purely local because Bob will never get a signal from Alice once she's decided the link is dead. A clock is commonly used for the heuristic, but care is needed. For example, if Bob is heavily loaded but Alice is not, she might set her timeout to be too short. If the situation is reversed, the timeout may be too long. An alternative is for Alice to count the events she receives on her other ports. She can declare the link dead if too many of these events are received before she gets one from Bob. This heuristic is effectively averaging over the workload of all the nodes connected to her, providing a more consistent metric.

1.6 The Conveyor Belt and Quantum Mechanics

From ./AE-Specifications-ETH/standalone/Conveyor-Belt.tex

Let's use Kevin's **conveyor belt metaphor** to describe time and its behavior under special relativity, general relativity, and contrast it with quantum mechanics and indefinite causal order.

1.6.1 Conveyor Belt as Time (Special and General Relativity)

Imagine time as a conveyor belt moving in one direction—forward. Objects, people, and events sit on this belt, carried steadily from past to future. The speed of the belt is consistent for everyone in classical physics. However, in **special relativity**, the speed at which individuals experience this conveyor belt can vary depending on how fast they are moving. If you're moving quickly relative to someone else, your conveyor belt slows down relative to theirs. You're still moving forward on your own belt, but the difference in speeds between the belts means time passes more slowly for you (this is time dilation).

In **general relativity**, gravity also affects the conveyor belt's speed. The closer you are to a massive object, the slower your conveyor belt moves compared to someone farther away. This is due to gravitational time dilation. Each person is on their own conveyor belt of time, but the rate of movement can change depending on their proximity to mass or their velocity. However, no matter the speed or how warped the conveyor belt becomes, time still moves consistently forward for each individual, even if it does so at different rates.

1.6.2 The Conveyor Belt and Quantum Mechanics

Quantum mechanics disrupts this conveyor belt metaphor. In the quantum world, things don't behave in a neatly predictable way, and time doesn't always behave like a simple forward-moving belt. Quantum systems can exist in superpositions, meaning they are in multiple states at once. If you try to apply the conveyor belt metaphor here, you'd have to imagine a belt where objects are not just moving forward but might also exist at multiple points along the belt at the same time. The simple idea of one thing following another breaks down because quantum mechanics deals with probabilities rather than certainties.

Moreover, events at the quantum scale don't always happen in a definite order. This leads to the concept of **indefinite causal order**.

1.6.3 Why the Conveyor Belt Breaks with Indefinite Causal Order

In indefinite causal order, the very idea of a consistent forward-moving belt doesn't hold. Imagine you're now dealing with multiple conveyor belts, but these belts can become entangled and swap positions. A key feature of indefinite causal order is that it is impossible to say whether event A happens before or after event B with certainty.

To continue the metaphor:

- Instead of a single conveyor belt that everyone rides, imagine a conveyor belt that can fork, loop, or even reverse for some events. On this belt, two events (say, picking up two different objects) might not have a clear sequence. You could pick up Object A before Object B, or Object B before Object A, or some mixture of both where the order isn't fixed until you observe it.

In **quantum mechanics**, particularly with **indefinite causal order**, time isn't a well-defined forward flow. Events can occur in a superposition of orders, defying the classical notion of a consistent timeline. There's no global, shared belt moving forward—some events might not even have a clear "before" and "after."

Special and General Relativity Work with the Conveyor Belt:

- **Special Relativity:** The belts are still mostly linear but stretch or compress based on speed, meaning time passes differently for observers in motion relative to one another.
- **General Relativity:** The belts warp under gravity, slowing down near massive objects but still remaining orderly, with a clear flow from past to future.

Indefinite Causal Order Breaks the Metaphor:

- In quantum mechanics, particularly in scenarios involving **indefinite causal order**, the metaphor falls apart. The belts aren't just stretching or warping—they no longer move in a straightforward, fixed order. Time becomes a superposition of possible orders, and the concept of a simple, directional flow of events can no longer apply.

The conveyor belt metaphor works well for **special** and **general relativity** because it reflects the relative nature of time: it still flows in one direction but at varying rates. For **quantum mechanics** and **indefinite causal order**, the metaphor breaks down, as time and causality become fluid, probabilistic, and not bound to a definite sequence.

1.6.4 Indefinite Causal Order

Indefinite causal order is a concept that comes from quantum mechanics and challenges the traditional notion that events occur in a definite sequence. In classical physics and most of computer science, events are understood to happen in a fixed, linear order—first A, then B, then C. However, indefinite causal order allows for scenarios where the order of events is not set in stone, meaning that, in certain quantum processes, it's impossible to say definitively whether A happened before B or vice versa.

When applied to computer science, particularly in areas like distributed systems, quantum computing, or complex event processing, indefinite causal order could have profound implications. Here's how it connects:

Distributed Systems:

In classical distributed systems, events are often coordinated using synchronized clocks or timestamps (like in Lamport clocks) to establish the order of events. But this assumes a definite, forward-in-time progression. In a world where causality can be indefinite, such as in quantum communication protocols, the assumption that events occur in a strict order breaks down. This could affect how we model consistency, causality, and concurrency in distributed systems.

For example, most distributed systems assume that causal relationships between events can be traced back in time (e.g., message A must have been sent before message B). Indefinite causal order could complicate this by introducing scenarios where it's unclear in what order events happened, leading to new ways to think about synchronization, coordination, and consistency.

Quantum Computing:

In quantum computing, the idea of indefinite causal order directly translates to certain computational advantages. One prominent example is the *Quantum SWITCH*, where two operations are performed, but their order is determined by the quantum state. This can lead to more efficient algorithms because the system doesn't need to follow a strict causal sequence of operations. For example, in some cases, tasks can be performed more efficiently by allowing operations to exist in a superposition of different causal orders.

Event-Driven Systems and Reactive Programming:

In event-driven or reactive systems, we often deal with streams of events and have to react to them in a specific order. If indefinite causal order were applicable, it would mean rethinking how systems react to events because the sequence of those events might not be fixed. This might lead to more flexible systems, but also requires new models of computation that can handle ambiguity in event timing and ordering.

Logical Time and Causal Models:

One key area where indefinite causal order could be explored in computer science is in extending the concept of "logical time" used in distributed systems. Today, we have models like vector clocks and Lamport clocks to track causal relationships between events. If we consider indefinite causal order, it might require developing new abstractions of time that are capable of representing ambiguous or superposed causal sequences. This could impact algorithms that rely on strict ordering, like consensus algorithms or conflict resolution mechanisms.

Future Implications:

Incorporating indefinite causal order into computer science, particularly in quantum computing and communication systems, might challenge foundational assumptions about how programs execute, how data is shared, and how systems coordinate. Researchers are beginning to explore how these ideas could lead to new architectures that embrace uncertainty or non-linearity in causality, pushing beyond the limits of classical synchronization methods.

1.6.5 Compare to Lingua Franca

Edward Lee's *Lingua Franca* (LF) and the *Reactors* programming model are built around very different assumptions about time and causality than those implied by *indefinite causal order* in quantum mechanics. To explore the comparison, let's break down how each framework approaches time and causality and where they diverge.

1. Lingua Franca and Reactors: Assumptions About Time

- **Deterministic Logical Time:**
  Lingua Franca (LF) is designed to deal with time explicitly in distributed systems and cyber-physical systems (CPS). Its model assumes **deterministic logical time**, meaning the sequence of events is well-defined and unambiguous. The idea is to provide a clean, deterministic model where events are processed according to a well-ordered timeline.
  In the *Reactors* programming model, time is also central. Events are triggered in response to other events based on strict logical dependencies and causal relationships. Reactors operate under a causality principle, where one event triggers another, creating a deterministic flow of information. LF provides a way to manage this using **timestamps**, ensuring that events execute in a known order, even in distributed systems where physical time (real-world clock time) may vary.
  This focus on **logical determinism** means that in LF, all participants in a distributed system have a consistent understanding of the event ordering, even if they are physically separated. The system explicitly synchronizes on logical time to ensure causal relationships are respected.

2. Indefinite Causal Order:

- **Causal Ambiguity:**
  Indefinite causal order is fundamentally different. In quantum mechanics, particularly in processes like the *Quantum SWITCH*, events can occur in a superposition of orders. There is no clear distinction between "before" and "after" for certain events. This is because quantum mechanics allows for a superposition of states, which can include superpositions of different causal orders. As a result, the causal relationships between events can be indefinite or non-deterministic.
  In the framework of indefinite causal order, the assumption of **deterministic logical time** breaks down. The sequence of events might not be clearly defined until some measurement or interaction occurs, and the system could be in a state where causality itself is undefined or ambiguous. This introduces a fundamental uncertainty about which events caused others, contrary to the deterministic approach taken by LF.

3. Comparing Assumptions on Time and Causality:

- **Lingua Franca** assumes that:
  - Time is deterministic, meaning every event has a clear cause and effect, and that logical time is the primary tool to ensure consistency in distributed systems.
  - Logical causality can always be enforced and preserved through careful design of time-triggered reactions and event-driven computations.
  - The real world might have uncertainties in physical clocks (due to relativity or synchronization issues), but **logical time** can compensate for those discrepancies and create a shared temporal framework that all parts of the system agree upon.
- **Indefinite Causal Order**, in contrast:
  - Rejects the idea that all events can be ordered definitively. There may be no clear "cause" or "effect" in certain quantum processes.
  - Embraces the possibility that causality itself can be indeterminate, allowing for events to be in superpositions of different causal orders.
  - Cannot be modeled with strict logical timestamps because the very concept of time order may be undefined until measurement occurs.

4. Where Lingua Franca Works and Where it Struggles:

- **Lingua Franca's conveyor belt analogy:**
  In LF, time is much like a carefully managed conveyor belt. Events flow forward in time, and the system is designed to ensure that no matter how distributed or asynchronous the components are, they will agree on the causal order of events. This works beautifully in systems where deterministic behavior is crucial, like in CPS, real-time systems, or distributed computing where timing guarantees are needed.
- **Indefinite Causal Order breaks this model:**
  The assumption behind LF that all events can be deterministically ordered along a logical timeline fundamentally breaks down when applied to scenarios involving indefinite causal order. Quantum systems with indefinite causality cannot be modeled using LF's deterministic logical time, as there may not be any single "correct" timeline.

5. The Conveyor Belt vs. Forked Paths:

- **Lingua Franca:**
  The conveyor belt metaphor works well for LF. The belt may be fast or slow, but it always moves in a clear direction, and events are placed in a strict order on it. Logical time is like a regulating mechanism that ensures the belts of different agents are synchronized in the right sequence.
- **Indefinite Causal Order:**
  Instead of a simple conveyor belt, imagine a system where multiple belts can split, re-merge, and sometimes create loops. Events may occur in superpositions of these different belts. When you try to observe them, you might find that two events could have happened in either order, or perhaps simultaneously in different orders depending on how you measure.

6. Conclusion:

- **Lingua Franca and Reactors** are built on the classical assumptions of time: causality is strict, logical, and deterministic. Events can be ordered using logical time, and causality can always be respected across distributed systems. LF succeeds in providing a structured, well-defined temporal model that works in a wide range of practical applications.
- **Indefinite Causal Order** challenges the very foundation of these assumptions. In quantum systems with indefinite causality, events don't necessarily have a strict temporal order. Causality becomes probabilistic or even undefined until observed. This fundamentally

contradicts LF's reliance on determinism and logical timestamps. If LF were to be applied in a system where causal order is indefinite, the model would need to be rethought to accommodate the ambiguity and superposition of events.

The core difference between Edward Lee's Lingua Franca and quantum-based indefinite causal order lies in how they treat time: LF enforces strict, deterministic event ordering, while indefinite causal order introduces a probabilistic, non-linear flow of events where time and causality can be ambiguous and undefined.

1.6.6 Compare to Lamport Clocks

Lamport's notion of **logical time** and the concept of **indefinite causal order** represent two very different approaches to understanding time and causality in distributed systems and quantum mechanics, respectively. Let's compare these concepts in detail:

1. Lamport's Logical Time:

Lamport introduced **logical clocks** to address the problem of ordering events in distributed systems, where physical clocks cannot be perfectly synchronized due to the limits of speed, network latency, and other issues. Logical time doesn't rely on actual clock time but instead on the **happens-before relation**, which captures the causal relationships between events. The main features of Lamport's logical time are:

- **Happens-Before Relation ($\rightarrow$):**
  If event A causes event B (e.g., by sending a message), we say $A \rightarrow B$. This relation defines the causal structure in the system.
- **Event Ordering:**
  Every process maintains a logical clock. When an event occurs, it increments its clock and attaches this timestamp to any message sent. When another process receives the message, it updates its logical clock to reflect that the event has already happened, ensuring that causality is respected.
- **Total Ordering:**
  While logical time can establish a partial ordering of events based on causal relationships, Lamport's logical clocks do not give a complete global time ordering. However, vector clocks and other mechanisms can extend logical clocks to achieve more precise causal tracking.

The key idea is that Lamport's logical time helps enforce **causal consistency** across distributed systems, ensuring that events are ordered in a way that respects causal relationships between them. However, the system still assumes a definite sequence of events—either event A happens before B, or B happens before A.

2. Indefinite Causal Order:

Indefinite causal order, originating from quantum mechanics, breaks the classical assumption of definite event ordering. In certain quantum processes, events can exist in a **superposition of different causal orders**. This means:

- **No Definite Ordering:**
  Two events, A and B, can occur in a superposition of different sequences. It's not clear whether A happens before B or vice versa. Both possibilities can exist simultaneously until an observation is made.
- **Quantum SWITCH Example:**
  A quantum protocol like the *Quantum SWITCH* allows two operations to be performed in such a way that the order in which they occur is not definite. For example, A might influence B and vice versa, but the precise causal sequence is only determined probabilistically upon measurement.
- **Causal Superposition:**
  Indefinite causal order challenges the classical notion of time and causality by allowing for events that don't have a single, well-defined causal relationship. This differs radically from classical models like Lamport's logical time, where the goal is to create a definite order for every event based on causal relationships.

3. Comparison of Causality:

- **Lamport's Logical Time:**
  In distributed systems, **causality is explicit** and must be preserved. Event A either happens before or after event B, and Lamport's logical clocks enforce this by ensuring all processes agree on the causal order of events, even in the absence of synchronized physical clocks. The aim is to create a consistent and well-defined timeline.
- **Indefinite Causal Order:**
  In quantum mechanics, **causality can be indefinite**, and events can exist in a superposition of causal orders. This is fundamentally different from Lamport's approach, where causality is strict and must be maintained. In quantum systems, there may not be a clear, observable sequence of events until they are measured.

4. Time and Event Ordering:

- **Lamport's Logical Time:**
  - Relies on the happens-before relation to **preserve a well-defined, consistent causal order** between events.
  - **Logical timestamps** are assigned to events based on causal dependencies, so we can always say A happened before B, or B

happened before A.
  - The focus is on ensuring causal consistency, even in asynchronous, distributed systems.
- **Indefinite Causal Order:**
  - Time is not well-defined. Events can occur in a **superposition** of causal orders, meaning A and B might have happened in both orders simultaneously, and there is no definite causal relationship until a measurement is made.
  - The classical notion of time, where events occur in a linear, ordered sequence, doesn't apply. Instead, causality becomes **probabilistic** and only resolves upon observation.
  - This contrasts with logical time, which **enforces order** regardless of real-time discrepancies.

5. Applications in Distributed Systems vs. Quantum Systems:

- **Lamport's Logical Time:**
  Works well in distributed systems where ensuring a consistent, agreed-upon event ordering is critical. It helps manage concurrency, ensure that messages and actions are causally related, and maintain consistency in asynchronous environments. Lamport's clocks can provide a **deterministic causal structure** in environments where physical time cannot be relied upon.
- **Indefinite Causal Order:**
  Applies to quantum systems where the classical notion of causality and time does not hold. This idea is particularly useful in **quantum communication** and **quantum computing**, where indefinite order can be exploited for computational advantages (e.g., the *Quantum SWITCH*). It introduces a level of **uncertainty and flexibility** that cannot exist in systems governed by logical time.

6. Fundamental Differences in Handling Time:

- **Lamport's Logical Time:**
  Assumes a **linear, forward-moving flow** of events, where causal relationships are always definite and traceable. Time, in this context, is deterministic, even though it may be logical rather than physical.
- **Indefinite Causal Order:**
  Time can be **non-linear and non-deterministic**. Events may not have a clear before/after relationship until they are observed, meaning that causality is not well-defined in the classical sense. Time is less a forward-moving arrow and more a **probabilistic, superposed system**.

7. Summary:

- **Lamport's Logical Time** provides a **definite ordering of events** in distributed systems, ensuring that causality is preserved and that events follow a strict happens-before relationship. It operates within a classical, deterministic framework, where every event has a well-defined place in the timeline.
- **Indefinite Causal Order** introduces the possibility that causality is not always fixed. In certain quantum systems, events may not have a clear ordering, and time is **probabilistic** rather than deterministic. This concept challenges the very foundation on which logical time is built, as it allows for causal ambiguity and non-determinism.

In short, Lamport's logical time is a deterministic tool designed to enforce causality in distributed systems, whereas indefinite causal order belongs to the realm of quantum mechanics, where causality itself can be in superposition. The two are fundamentally at odds, as one imposes strict order while the other allows for causal uncertainty.

1.7 The Illusion of Simultaneity with Perfect Atomic Clocks

From ./AE-Specifications-ETH/standalone/Andromeda.tex

The relativity of simultaneity implies that two observers moving relative to one another can disagree on what events are happening "right now" at distant locations. Even if both observers carry perfectly synchronized atomic clocks, their determinations of simultaneous events at remote locations (e.g., the Andromeda Galaxy) can differ dramatically.

1.7.1 Time Shift Due to Relative Motion

Let v be the relative velocity between two observers, and D the distance to a distant object (e.g., a galaxy). The relativity of simultaneity predicts a difference in the perceived "now" at the remote location:

$$\Delta t \approx \frac{vD}{c^2}, \tag{1.1}$$

where c is the speed of light.

Examples

- For walking speed ($v = 1.39\,\mathrm{m/s}$) and $D = 2.5$ million light-years (Andromeda):

$$\Delta t \approx \frac{1.39 \times 2.365 \times 10^{22}}{(3 \times 10^8)^2} \approx 4.2 \text{ days}$$

- For hypersonic flight ($v = 1700\,\mathrm{m/s}$):

$$\Delta t \approx \frac{1700 \times 2.365 \times 10^{22}}{(3 \times 10^8)^2} \approx 5170 \text{ days} \approx 14 \text{ years}$$

1.7.2 Atomic Clock Precision

Modern optical lattice clocks can achieve stability better than 10^{-18}, corresponding to an error of less than 1 second over 30 billion years. Over a day (86400 s):

$$\delta t_{\text{clock}} = 10^{-18} \times 86400 \approx 8.64 \times 10^{-14} \text{ seconds} \tag{1.2}$$

THIS IS UTTERLY WRONG

It mistakes Gaussian variations in noise for time intervals. Mathematics doesn't work this way, The problem is (with $t \pm \delta t$) is not an interval. It's a belief in absolute simultaneity.

This is less than a femtosecond, utterly negligible compared to the relativity-induced differences in simultaneity.

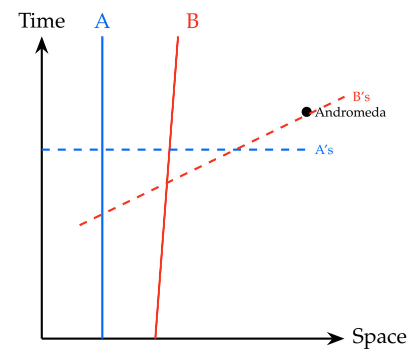

Figure 1.2: Two observers disagree on simultaneity with Andromeda events.

1.7.3 Diagram: Disagreement Despite Synchronized Clocks

No degree of 'precision' or 'disciplining' of clocks will enable us to 'synchronize time".

Simultaneity is impossible in theory. It will therefore be problematic in practice. Physicists know this. Computer scientists have yet to discover relativity and quantum mechanics.

1.7.4 Conclusion

Perfectly synchronized atomic clocks do not resolve disagreements about simultaneity at distant locations. Such disagreements are a feature of spacetime itself in special relativity — not of timekeeping imprecision.

1.8 Zero: The Natural Number that Isn't

From commented out text in ./AE-Specifcations-ETH/standalone/Zero.tex

Zero is arguably the most fascinating number in mathematics, occupying a uniquely paradoxical position in our number systems. It serves as both a placeholder and a quantity in its own right, a concept that revolutionized mathematics when fully developed. Yet despite its fundamental importance, zero's classification remains a point of contention, particularly regarding whether it belongs among the natural numbers.

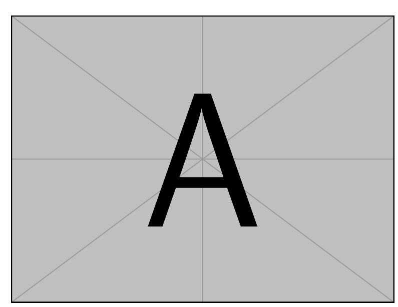

Figure 1.3: Various historical representations of zero across civilizations.

The question of zero's status as a natural number isn't merely semantic—it reflects deeper mathematical principles and has practical implications for how we define foundational concepts.

1.9 Historical Development

The concept of zero emerged independently in several ancient civilizations. The Babylonians used a placeholder symbol for zero around 300 BCE, while the Mayans developed a complete zero symbol around 350 CE. However, it was in India where zero was first treated as a proper number, with Brahmagupta establishing formal arithmetic rules for zero in the 7th century CE.

Brahmagupta's rules included defining $a - a = 0$ and $a \times 0 = 0$, but he struggled with division by zero, a problem that continues to challenge students today.

Mathematicians throughout history have disagreed about zero's classification. The ancient Greeks, whose work heavily influenced Western mathematics, had no concept of zero as a number. When European mathematics eventually incorporated zero, disagreements about its nature persisted.

1.10 Set-Theoretic Foundations

In modern mathematics, natural numbers are typically defined using set theory. The two predominant approaches yield different results regarding zero:

1. **Von Neumann ordinals**: Define natural numbers recursively where $0 = \emptyset$ (the empty set), $1 = \{0\} = \{\emptyset\}$, $2 = \{0, 1\} = \{\emptyset, \{\emptyset\}\}$, and so on. In this construction, zero is the first natural number.
2. **Zermelo ordinals**: Define 1 as $\{\emptyset\}$, 2 as $\{\{\emptyset\}\}$, etc. Here, counting begins at 1, excluding zero from the natural numbers.

This fundamental distinction reflects different mathematical perspectives on what counting means.

The choice between these constructions reveals a fundamental question: Is mathematics primarily about counting (starting at 1) or about formal structures (where starting at 0 offers advantages)?

1.11 Mathematical Arguments

Several compelling mathematical arguments support excluding zero from the natural numbers:

1. **Etymology and intuition**: "Natural numbers" traditionally refer to the counting numbers used in everyday life. When counting objects, we begin with one, not zero.
2. **Multiplicative properties**: The natural numbers form a monoid under multiplication with identity element 1. Including zero breaks this structure since zero lacks a multiplicative inverse.
3. **Division**: In the natural numbers excluding zero, division (when defined) always yields a unique result. Including zero introduces complications, as division by zero is undefined.

4. **Induction principle**: Mathematical induction typically starts with a base case of $n = 1$, implicitly excluding zero from consideration.

1.12 Notational Clarity

To avoid ambiguity, mathematicians have developed notational conventions:

- $\mathbb{N}$ or $\mathbb{N}_1$: Natural numbers starting from 1
- $\mathbb{N}_0$ or $\mathbb{N} \cup \{0\}$: Natural numbers including zero

1.13 Practical Implications

Whether zero is included among the natural numbers has substantial implications in various mathematical contexts:

- In combinatorics, excluding zero aligns with counting principles (you can't have zero of something when counting discrete objects)
- In number theory, including zero simplifies many formulations
- In computer science, zero-based indexing (starting array indices at 0) has proven advantageous for algorithm implementation

1.14 The Deeper Meaning

The debate about zero's status reveals a profound truth: mathematical classifications aren't discovered in nature but constructed by humans based on utility and consistency. The question "Is zero a natural number?" ultimately depends on the mathematical context and purpose.

This ambiguity isn't a flaw in mathematics but a reflection of its adaptability. Mathematical structures can be defined in different ways to serve different purposes, and these definitions are judged by their usefulness and elegance rather than absolute correctness.

Zero remains the bridge between positive and negative numbers, neither fully belonging to either domain yet essential to both. Perhaps this liminal position is precisely what makes zero so mathematically powerful—it stands at the boundary, connecting different mathematical realms.

The natural numbers may have begun as simple counting tools, but mathematics has evolved into a sophisticated framework where even our most basic numerical concepts reveal surprising complexity. Zero's contested status reminds us that mathematics, despite its reputation for certainty, contains fundamental questions whose answers depend on perspective.

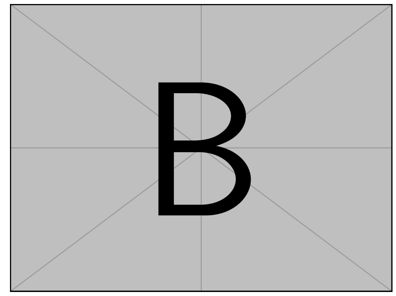

Figure 1.4: Visualization of the multiplicative monoid structure of natural numbers without zero.

The International Standards Organization (ISO 80000-2) defines $\mathbb{N}$ as starting from 1, while many computer scientists and set theorists prefer to include 0.

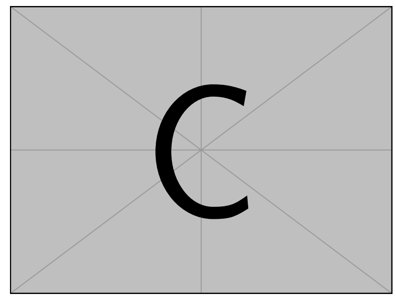

Figure 1.5: Example of zero-based indexing in computer programming.