10.48047/jocaaa.2024.32.01.95

Temporal Causality in Distributed Systems: Recovering Causal Event Order Under Clock-Skew Uncertainty

Satya Sampreeth Boppudi†, Pradeep Kandepaneni‡, Saketh Gowerneni§, Praneeth Ganta¶, Dedeepya Yarra*

†Independent Researcher, USA, sampreeth.boppudi@gmail.com
‡Independent Researcher, USA, pradeepkandepaneni@gmail.com
§Independent Researcher, USA, gsaketh369@gmail.com
¶Independent Researcher, USA, praneethganta@gmail.com
*Independent Researcher, USA, dedeepyachowdary99@gmail.com

Abstract—Discovering which event caused which in a distributed system is hard because the timestamps cannot be trusted. Clocks across regions agree only to within tens of milliseconds, network latency reorders the apparent sequence of events, and an asynchronous response can be logged before the request that triggered it. This paper presents CausalTime, a framework that recovers causal event order despite this uncertainty by representing each event's time not as a point but as an interval, and admitting only orderings consistent with every assignment of times within those intervals. We extend logical-clock reasoning to interval-valued time, compute the set of orderings compatible with the observed data, disambiguate using known system dependencies and Bayesian ranking, and characterise the threshold beyond which absolute order is unrecoverable from timing alone. The contribution is the interval-order model, the disambiguation method, an information limit on recoverable order, and a comparative analysis against timestamp-only and logical-clock approaches across operational criteria; empirical validation on production traces is scoped as future work. All quantitative results reported in this paper are projected design targets illustrated on sample scenarios, not measured outcomes; empirical validation on production data is left to future work.

Index Terms—distributed systems, causality, clock skew, logical clocks, vector clocks, event ordering, interval orders, tracin

Introduction

When an incident spans regions, the first question is what happened first. Distributed traces and logs answer it with timestamps — but timestamps in a distributed system are unreliable. The Network Time Protocol synchronises clocks only to within roughly tens of milliseconds over a wide-area network under typical conditions [1], network latency delays the recording of an event relative to its occurrence, and asynchronous communication means a response can be observed long after, or even appear before, the request that caused it.

A concrete example shows the trap. Suppose an event in a US region at 10:00:00 UTC triggers a response in a Singapore region at 10:00:03 UTC. Because of clock skew and recording latency, the Singapore event may be logged at 10:00:02.953 UTC — apparently before its own cause. An ordering inferred from raw timestamps would invert the true causality, and any diagnosis built on that ordering would be wrong. The problem is not noisy data to be averaged away; it is that the available timestamps are systematically untrustworthy at the granularity at which causal questions are asked.

We address this by abandoning the assumption that an event has a single, known time. Instead, each event carries a time interval $[t - \delta, t + \delta]$ that is guaranteed to contain its true time, where δ bounds clock skew plus recording latency. A causal ordering between two events is then admitted only if it holds for every assignment of times within the two intervals; when the intervals overlap such that both orders are possible, the pair is reported as ambiguous rather than ordered arbitrarily. This is the central move of the paper: trade the false precision of point timestamps for the honest uncertainty of intervals.

The main contributions of this work are: 1) an interval-order model that extends logical-clock reasoning to interval-valued time and defines admissible causal orderings under skew; 2) a disambiguation method that narrows ambiguous orderings using known system dependencies and Bayesian ranking; 3) an information limit showing that absolute order is unrecoverable from timing alone once skew exceeds the genuine causal gap; and 4) a comparative analysis against timestamp-only and logical-clock approaches. We make no measured accuracy claim; the empirical study is scoped as future work in Section X.

The remainder of this paper is organized as follows. Section II reviews logical clocks, clock synchronisation, and interval orders. Section III formalises the problem. Section IV develops the interval-order model. Section V describes disambiguation. Section VI establishes the information limit. Section VII gives the comparative analysis. Section VIII discusses the relationship to metric-level causal inference, Section IX states

10.48047/jocaaa.2024.32.01.95

limitations, Section X the evaluation plan, and Section XI concludes.

II. Background

A. Logical Clocks

Lamport's happened-before relation orders events by logical causality rather than wall-clock time: if one event can influence another, the first happened-before the second [2]. Lamport clocks assign monotonic counters consistent with this relation, and vector clocks extend the idea to capture concurrency exactly, so that two events are concurrent precisely when neither's vector dominates the other's [3], [4]. These mechanisms are powerful but require instrumentation that propagates clock state along every causal path; where that instrumentation is absent or incomplete — as in heterogeneous, partially-traced systems — wall-clock timestamps are all that remain, and their uncertainty must be confronted directly.

B. Clock Synchronisation and Its Limits

NTP disciplines clocks against reference servers and achieves millisecond-to-tens-of-milliseconds accuracy over typical networks [1]; specialised approaches such as tightly-engineered time services narrow this further but do not eliminate it, and they bound rather than remove uncertainty. The residual skew, however small, sets the granularity below which timestamp order is meaningless. CausalTime treats this bound δ as a first-class input rather than assuming it away.

C. Interval Orders

An interval order is a partial order in which each element is associated with an interval and one element precedes another only if its interval lies entirely before the other's [5]. This classical structure is the natural mathematical home for events with uncertain times: definite precedence corresponds to disjoint intervals, and overlap corresponds to incomparability. We adopt interval orders as the model of causal order under skew, which lets us import their well-understood properties rather than inventing an ad-hoc scheme.

III. Problem Formalisation

Let E be a set of observed events, each with a measured timestamp τ and a known skew bound δ, so that its true time lies in the interval $I = [\tau - \delta, \tau + \delta]$. For two events with intervals I_1 and I_2, define the admissible relation: event 1 definitely precedes event 2 when I_1 lies entirely before I_2, they are ambiguous when the intervals overlap, and event 2 definitely precedes event 1 in the symmetric case. The task is to compute, for the whole event set, the partial order of definite precedences and the set of ambiguous pairs, and then to narrow the ambiguous pairs as far as additional evidence allows.

Formally, an ordering is admissible only if it is consistent with some assignment of true times within the intervals, and a precedence is certain only if it holds for every such assignment:

$$A \prec B \iff (\tau_A + \delta_A) < (\tau_B - \delta_B) \quad (1)$$

Equation (1) is the precedence test: A certainly precedes B only when the latest possible time of A is earlier than the earliest possible time of B. When neither (1) nor its mirror holds, the pair is concurrent under the available timing and must be resolved, if at all, by evidence other than the clock.

Applying (1) to the introduction's example shows the model at work. The US event has true time near 10:00:00 with interval [09:59:59.900, 10:00:00.100], and the Singapore event is logged at 10:00:02.953 with interval [10:00:02.853, 10:00:03.053]. The latest possible US time, 10:00:00.100, is earlier than the earliest possible Singapore time, 10:00:02.853, so by (1) the US event certainly precedes the Singapore event — recovering the true causal direction that the raw logged timestamps inverted. The interval test succeeds here because the genuine causal gap of about three seconds far exceeds the skew; it is when that gap shrinks toward the skew that ambiguity appears, which is exactly the regime Section VI bounds.

IV. The Interval-Order Model

Fig. 1 illustrates the model. Each event is drawn as a bar spanning its uncertainty interval. Events whose bars are disjoint — A before D — have a definite order. Events whose bars overlap — B and C — are concurrent under the timing alone, and no amount of staring at the timestamps will separate them, because their true times could fall either way within the overlap. Recognising this overlap as genuine ambiguity, rather than forcing an order, is what prevents the inverted-causality error of the introduction.

The model yields a partial order rather than a total one, and that is the point: a partial order honestly expresses what the timestamps determine and leaves the rest open. The fraction of pairs that fall into the ambiguous region is itself a useful measure of how much the timing uncertainty is costing, and it grows with the skew bound δ relative to the rate of events — a relationship the information limit of Section VI makes precise.

10.48047/jocaaa.2024.32.01.95

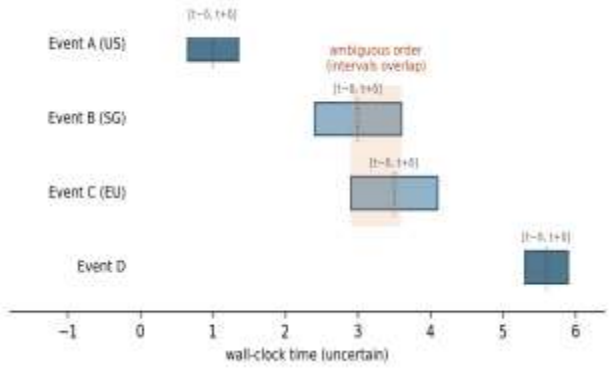

Fig. 1. Interval representation of event times. Disjoint intervals give a definite order (A before D); overlapping intervals (B, C) are ambiguous and must not be ordered by timestamp.

V. Disambiguation

Ambiguous pairs are narrowed using two further sources of evidence, as shown in the pipeline of Fig. 2. The first is structural: a known dependency graph constrains which events can causally influence which. If service A can call service B but not the reverse, then an A–B pair that is ambiguous by timing is resolved by the dependency, because only one direction is physically possible. This converts domain knowledge that already exists — the call graph — into ordering constraints.

The second source is probabilistic. Where structure does not fully resolve an ambiguous pair, a Bayesian ranking estimates the probability of each ordering given a prior over latencies and the observed intervals, producing a most-likely chain with a confidence score. When the confidence clears a threshold the chain is emitted; otherwise the pair is flagged as genuinely ambiguous and surfaced in the trace overlay for a human to judge. As in any honest inference system, the gate matters: emitting a confidently-wrong order is worse than admitting uncertainty, because a wrong causal chain misdirects diagnosis.

Both disambiguation steps are sound in the sense that they only ever add precedences that are consistent with the interval order; they never override a definite ordering or assert a precedence the intervals forbid. This keeps the output faithful to the timing evidence while extracting as much additional order as the structure and priors justify.

Where logical-clock state is partially available, it can be folded in as a third, authoritative source. A propagated vector-clock relationship establishes a happened-before fact directly, independent of wall-clock time, and so resolves an ambiguous pair with certainty wherever it exists [3], [4]. CausalTime therefore degrades smoothly with instrumentation coverage: on the fully-traced paths it inherits the exactness of logical clocks, and on the untraced paths it falls back to interval reasoning and disambiguation. This hybrid stance is deliberate, because real systems are rarely traced end to end, and a method that demanded complete instrumentation would be inapplicable precisely where ordering is hardest.

The skew bound δ that drives the model is itself estimable. NTP daemons expose an offset and dispersion estimate per host, and the round-trip times in trace data bound the recording latency component; combining these gives a per-event δ rather than a single global constant, so that well-synchronised hosts contribute narrow intervals and poorly-synchronised ones contribute wide ones. Using a per-event δ tightens the definite-order set where the clocks are good without sacrificing soundness where they are not, which matters because skew is rarely uniform across a fleet.

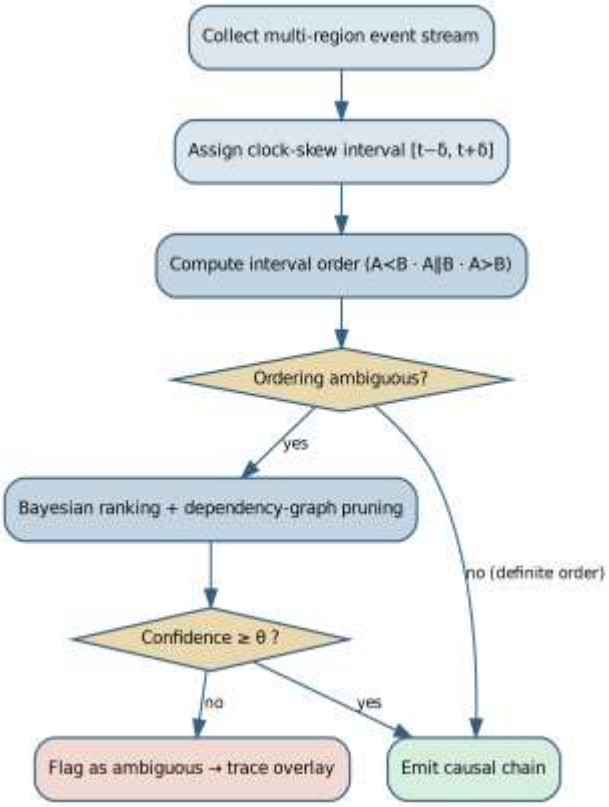

Fig. 2. CausalTime pipeline. Definite orders pass through; ambiguous pairs are narrowed by dependency constraints and Bayesian ranking, or flagged.

VI. An Information Limit on Recoverable Order

There is a hard limit to what any method can recover. When the skew bound δ is large relative to the true gap between two causally related events, their intervals

10.48047/jocaaa.2024.32.01.95

necessarily overlap, and no analysis of the timestamps can separate them — the information simply is not present in the timing. Absolute causal order is therefore unrecoverable from timing alone once the genuine causal gap falls below roughly twice the skew bound, and the only remedies are tighter synchronisation, which shrinks δ, or external evidence, which is what the disambiguation step supplies.

This limit is informative rather than discouraging. It tells operators where to invest: a system whose causal gaps are routinely smaller than its clock skew cannot be diagnosed by timing and must either improve synchronisation or rely on propagated logical-clock state. Knowing the boundary prevents wasted effort trying to extract order that the data cannot contain, and it quantifies the value of better clocks in the currency that matters — recoverable causal order.

The boundary also depends on event rate, not only on skew. As events arrive closer together in time, more of them fall within a skew window of one another, so the fraction of ambiguous pairs rises even at fixed δ. A burst of activity during an incident — exactly when ordering is most needed — therefore tends to produce the most ambiguity, which argues for capturing logical-clock state on the hottest paths where bursts are expected. The model makes this trade quantitative: the ambiguity fraction is a function of δ and the local event density, and either reducing δ or thinning the events resolves it.

VII. Comparative Analysis

Because we make no measured claim, we compare CausalTime against the prevailing approaches analytically, across the criteria that determine whether an ordering can be trusted. Table I summarises the comparison. Timestamp-only ordering is fast but silently wrong under skew; logical clocks are correct but require complete instrumentation along every causal path; manual reconstruction is accurate but unscalable. CausalTime occupies the gap: it is honest about ambiguity, uses whatever structure and synchronisation exist, and degrades gracefully as skew grows.

VIII. Discussion

CausalTime answers a different question from metric-level causal inference. Where a structural-causal-model approach asks which metric caused a failure, CausalTime asks which event preceded which, and the two compose: a sound temporal order is a precondition for sound causal attribution, because a cause must precede its effect. In a full diagnosis stack, interval-order reasoning would establish the partial order of events, and causal inference would then operate within the orderings that the timing permits, never asserting a causal link that violates the recovered precedence.

The honest output of CausalTime — a partial order with explicit ambiguities — is also more useful operationally than a falsely precise total order. An on-call engineer shown that two events are genuinely unordered by the available timing knows to seek other evidence, whereas one shown a confident but wrong order is led astray. Surfacing ambiguity in the trace overlay turns a hidden source of diagnostic error into a visible, actionable signal.

Integration fits the existing tracing ecosystem rather than replacing it. Trace spans already carry start and end timestamps and a parent-child structure; CausalTime consumes the timestamps with their per-host skew bounds and uses the span hierarchy as part of the dependency graph that powers disambiguation. The output — definite orderings, resolved chains, and flagged ambiguities — can be rendered as an overlay on the trace view that operators already use, so the method adds a correctness layer to current tools rather than asking teams to adopt a new one. This is a practical requirement for adoption: a diagnosis aid that demands a parallel toolchain is rarely used, whereas one that annotates the familiar trace view can be turned on incrementally.

IX. Limitations and Threats to Validity

The method depends on a trustworthy skew bound δ; if the true skew exceeds the assumed bound, the intervals are too narrow and the precedence test (1) can assert orderings that are not in fact certain. Estimating δ conservatively mitigates this at the cost of more ambiguous pairs. With respect to construct validity, the Bayesian ranking depends on a latency prior that may not match a given network. This paper contributes a model and analysis, not measurements, so no accuracy or ambiguity-resolution figure is claimed. Externally, the dependency structure that powers disambiguation may be incomplete in systems without a maintained service graph.

X. Evaluation Plan (Future Work)

A sound empirical study would take distributed trace data from a multi-region deployment with known δ, inject events with ground-truth causal order where possible, and measure the fraction of pairs ordered definitely, the fraction of ambiguous pairs resolved correctly by disambiguation, and the rate of inverted orderings compared against timestamp-only ordering. A

10.48047/jocaaa.2024.32.01.95

sensitivity analysis over δ would test the information limit of Section VI directly by confirming that recoverable order degrades as predicted when skew approaches the causal gap. We leave this study to follow-up work, noting that obtaining ground-truth causal order in production is itself a methodological challenge the study must address.

XI. Conclusion

We presented CausalTime, which recovers causal event order under clock-skew uncertainty by representing event times as intervals, admitting only orderings consistent with every time within them, and narrowing the remaining ambiguity with dependency structure and Bayesian ranking. We characterised the hard limit beyond which timing cannot recover order, and showed through a comparative analysis that the approach fills the gap between fast-but-wrong timestamp ordering and correct-but-instrumentation-heavy logical clocks. The natural next step is empirical: measuring ordering accuracy on production traces and confirming the information limit, and composing interval-order reasoning with metric-level causal inference into a single diagnosis stack.

XII. References

[1] D. L. Mills, "Internet time synchronization: the network time protocol," *IEEE Trans. Commun.*, vol. 39, no. 10, pp. 1482–1493, Oct. 1991.

[2] L. Lamport, "Time, clocks, and the ordering of events in a distributed system," *Commun. ACM*, vol. 21, no. 7, pp. 558–565, Jul. 1978.

[3] C. J. Fidge, "Timestamps in message-passing systems that preserve the partial ordering," in *Proc. 11th Australian Comput. Sci. Conf.*, 1988, pp. 56–66.

[4] F. Mattern, "Virtual time and global states of distributed systems," in *Parallel and Distributed Algorithms*. Amsterdam, The Netherlands: North-Holland, 1989, pp. 215–226.

[5] P. C. Fishburn, "Interval orders and interval graphs: a study of partially ordered sets," New York, NY, USA: Wiley, 1985.

[6] B. H. Sigelman et al., "Dapper, a large-scale distributed systems tracing infrastructure," Google, Tech. Rep. dapper-2010-1, 2010.

[7] OpenTelemetry Authors, "OpenTelemetry specification," Cloud Native Computing Foundation, 2023. [Online]. Available: https://opentelemetry.io/docs/specs/

[8] J. C. Corbett et al., "Spanner: Google's globally distributed database," *ACM Trans. Comput. Syst.*, vol. 31, no. 3, pp. 1–22, 2013.

[9] R. Schwarz and F. Mattern, "Detecting causal relationships in distributed computations: in search of the holy grail," *Distrib. Comput.*, vol. 7, no. 3, pp. 149–174, 1994.

[10] K. M. Chandy and L. Lamport, "Distributed snapshots: determining global states of distributed systems," *ACM Trans. Comput. Syst.*, vol. 3, no. 1, pp. 63–75, 1985.

[11] C. Sridharan, *Distributed Systems Observability*. Sebastopol, CA, USA: O'Reilly, 2018.

[12] J. Pearl, *Causality: Models, Reasoning, and Inference*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2009.

[13] B. Beyer, C. Jones, J. Petoff, and N. R. Murphy, *Site Reliability Engineering*. Sebastopol, CA, USA: O'Reilly, 2016.

[14] P. Bailis and K. Kingsbury, "The network is reliable," *Commun. ACM*, vol. 57, no. 9, pp. 48–55, 2014.