

A Formal Model for Forensically Sound Timestamp Normalization

Lorenz Hornung 

Open Universiteit, Heerlen, The Netherlands

Carly Bakker

Netherlands Forensic Institute, The Hague, The Netherlands

Bouke Timbermont

Open Universiteit, Heerlen, The Netherlands

Netherlands Forensic Institute, The Hague, The Netherlands

Roel van Dijk

Netherlands Forensic Institute, The Hague, The Netherlands

Harm van Beek 

Open Universiteit, Heerlen, The Netherlands

Netherlands Forensic Institute, The Hague, The Netherlands

Natasha Alechina 

Open Universiteit, Heerlen, The Netherlands

Utrecht University, The Netherlands

Abstract

Digital forensic analysts interpret timestamps from digital systems to assess what happened in the past. Since these systems encode time according to different underlying domains, timestamps cannot be interpreted uniformly. To enable cross-system analysis, timestamps must therefore be normalized. In this paper, we propose a formal model for forensically sound timestamp normalization and examine its properties. We show that normalization generally does not guarantee the comparability of normalized timestamps. This means that the temporal order of digitally tracked events cannot always be determined. Furthermore, we illustrate that existing forensic tools do not adhere to the proposed normalization, resulting in violations of forensic soundness in the example discussed.

2012 ACM Subject Classification Applied computing → Evidence collection, storage and analysis

Keywords and phrases Digital Forensics, Formal Model, Timestamp, Normalization

Digital Object Identifier 10.4230/OASICS.TIME.2026.10

1 Introduction

Law enforcement increasingly relies on digital investigations that analyze devices such as smartphones, computers, smartwatches, or servers. For these investigations, as in traditional forensics, traces¹ are relevant. Conceptually, digital traces are evidential artifacts derived from investigated digital sources that support the assessment of hypotheses (e.g., whether a suspect committed fraud or possessed illegal material) [11].

Important artifacts of this nature are so-called *timestamps*. Digital systems (such as file systems, operating systems, and many others) track when events occur and store this information as timestamps in databases, logs or file metadata [10, 11]. These timestamps are then used for event reconstruction, the investigative task of constructing and assessing hypotheses about past actions. Importantly, event reconstruction normally involves examining timestamps stored across devices and digital systems [9, 23].

¹ In forensic science, traces refer to signs of prior activity. This differs from traces in temporal logic and system verification, where it is defined as sequence of states or transitions (see, for example, [5]).



To enable cross-device and cross-system analysis, timestamps must be represented uniformly [9]. We refer to this task as *timestamp normalization* (informally defined in Section 2.2.3). Although forensic software, such as Plaso [35], Sleuth Kit [36], and Hansken [33], performs this operation and it is recognized as a challenge by digital forensic investigators [28], timestamp normalization for digital forensics has received no theoretical attention.

Contribution. This paper addresses this gap by providing a formal model of timestamp normalization that aligns with the principles of forensic soundness. This formalization establishes a standard that documents the essential properties to which timestamp normalization must adhere in digital forensic analysis. Upon examining the formalization, we show that forensically sound normalization does not guarantee the comparability of normalized timestamps. In other words, for certain timestamps, it is impossible to determine the order of tracked events. Additionally, we demonstrate that state-of-the-art forensic software inadequately normalizes timestamps, which might lead to forensically unsupported conclusions.

Outline. In Section 2, we introduce time tracking of civil time, time tracking in digital systems and how this results in distinct underlying time domains highlighting the need for normalization. Additionally, we informally define timestamp normalization and outline the forensic soundness requirements for it. In Section 3, we introduce the formal framework leading to the definition of normalization of timestamps (Section 3.6) building upon the definition of normalization of time domains (Section 3.3). We show how the formalization of normalization leads to incomparable timestamps in Section 3.5. In Section 4, we illustrate how Sleuth Kit, Plaso, and Hansken do not soundly normalize timestamps and elaborate on the resulting negative consequences using a real-world example. We discuss related work in Section 5 and outline directions for future research in Section 6. Proofs omitted throughout this paper are available in the appendix (Appendix A).

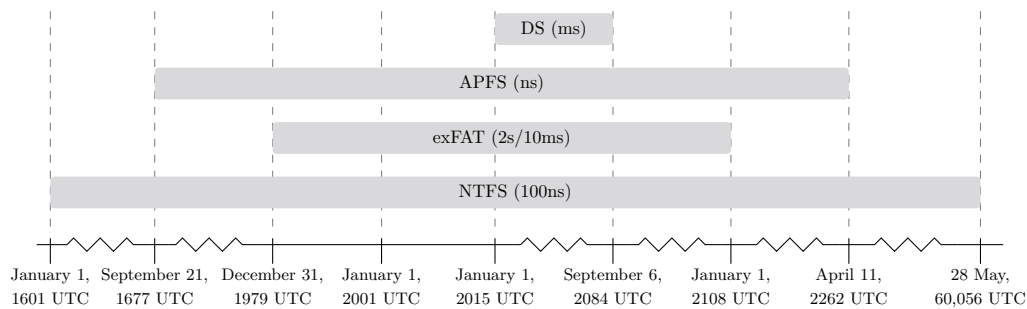
2 Defining Time in Digital Systems

Digital systems depend on defined time points to assign timestamps. We refer to the set of time points that a system can use to assign timestamps as its *time domain*. No single time domain is used by all digital systems; rather, the time domains of digital systems vary.

To illustrate this aspect, we will characterize the time domains of the following digital systems: the extensible File Allocation Table (exFAT) [29], the Apple File System (APFS) [3], the New Technology File System (NTFS) [30], and Discord Snowflake (DS) [18]. Figure 1 provides an overview over their time domains. ExFAT, APFS, and NTFS are file systems frequently found on portable storage devices (e.g., USB flash drives), Apple devices, and Windows devices, respectively. A file system organizes the storage and retrieval of files and directories on a storage device. In contrast, DS is a unique identifier containing a timestamp, used for messages, users, and servers on the social network Discord. We chose exFAT, APFS, and NTFS because they are significant in digital investigations, while DS serves as an example of a less common digital system that is nonetheless interesting in terms of timestamping.

2.1 Time Tracking of Civil Time

Civil life depends on a shared understanding of time. To ensure this, time standards are used which function as unified foundation to refer to points in time. For our purpose, the time standards Universal Time (UT1), International Atomic Time (TAI), and Coordinated Universal Time (UTC) are of interest. For references on UT1, TAI and UTC, see [24, 25]. Simplified, but sufficient for our purpose, these time standards relate as follows.



■ **Figure 1** Comparison of time domains of DS, APFS, exFAT, and NTFS.

TAI is the weighted average of hundreds of cesium atomic clocks spread worldwide and is based on fixed duration of a second (which is tracked by these atomic clocks). Based on this duration of a second, other units of time, like milliseconds, minutes or hours, are defined as fixed multiples of the second. For instance, 1/1000th of a second is a millisecond and 60 seconds are a minute. In contrast, Universal Time (UT1) tracks time by assessing the earth's rotation relative to the sun. Due to irregularities of earth's rotation, UT1 and TAI drift. UTC is the compromise of UT1 and TAI, as it keeps the same duration for seconds as TAI, but adds leap seconds when needed (at unpredictable times) to keep UTC within 0.9 seconds of UT1. Leap seconds are extra seconds added (or subtracted) at March 31, June 30, September 30, or December 31. So far, 27 leap seconds have been added and no leap second has been subtracted. As a consequence, some minutes of UTC consist of 61 instead of 60 seconds. For instance, the last leap second is December 31, 2016 23:59:60 UTC.

UTC is the primary time standard upon which all timezones are based. Local timezones are defined by an offset relative to UTC which can change throughout the year. For example Nepal uses Nepal Standard Time (NPT), defined as UTC+05:45, for the entire year. The Netherlands uses the Central European Time (CET) during winter, defined as UTC+01:00, and Central European Summer Time (CEST) during summer, defined as UTC+02:00.

2.2 Time Tracking in Digital Systems

Digital systems frequently, but not necessarily, use UTC as the foundation of their time domain. However, due to practical limitations UTC does not define their time domain; i.e., the finite set of time points that a system can represent as a timestamp.

2.2.1 Defining Time Points by Date and Time

Some digital systems base the encoding of timestamps on common human calendar systems. For instance, exFAT's time domain is based on the Gregorian calendar. It provides 16 bits for the date (7 bits for the year, 4 bits for the month, and 5 bits for the day), 16 bits for the time with a resolution² of 2 seconds (5 bits for hours, 6 bits for minutes, and 5 bits for seconds with a resolution of two seconds) and, only for file creation and last modification timestamps, 8 additional bits for milliseconds with a resolution of 10 milliseconds. For each group of bits, a range of values is defined. As an example, for years the range value is from 0

² In digital forensics, the notions "resolution", "granularity", and "precision" are used to refer, in the context of timestamps, to the smallest unit of time that a digital system can distinguish. To avoid confusion, we use only "resolution" to refer to this unit.

(year 1980) to 127 (year 2107), while for months the range value is from 1 (month January) to 12 (month December). The valid range for seconds is 0 to 29 and for milliseconds 0 to 199, illustrating that exFAT does not account for leap seconds.

Since exFAT provides these values relative to the local time (sensitive to timezone), it keeps track of the relative offset to UTC. This offset is given by 8 bits (6 bits signed decimal for offset with a resolution of 15 minutes, 1 bit to define whether offset is known). The valid range for the offset is -64 (UTC - 16:00) to 63 (UTC + 15:45) with 0 being UTC. Due to the offset of UTC, exFAT's time domains relative to UTC are defined as follows with D_{exFAT}^1 denoting the time domain for file creation and last modification timestamps, whereas D_{exFAT}^2 refers to the time domain for the last access timestamps.

- D_{exFAT}^1 : Every 10 milliseconds between December 31, 1979 00:08:00.00 UTC and January 1, 2108 15:45:59.99 UTC excluding leap seconds
- D_{exFAT}^2 : Every 2 seconds between December 31, 1979 00:08:00 UTC and January 1, 2108 15:45:58 UTC excluding leap seconds

2.2.2 Defining Time Points by Elapsed Time since Epoch

More frequently, time points are defined by counting forwards and, when applicable, backwards from a fixed point in time, called the epoch. The time domains of DS, APFS, and NTFS are defined by this method. As a foundation, they define the following epochs: January 1, 1970 00:00:00 UTC for APFS, January 1, 1601 00:00:00 UTC for NTFS, January 1, 2015 00:00:00 UTC for DS.

Counting forward and backward depends on a system's time resolution. Each step corresponds to one unit of the chosen resolution. APFS uses nanosecond resolution, NTFS uses 100-nanosecond resolution, and DS uses millisecond resolution.

Since these systems define a finite space allocated for storing the numeric value representing the counted elapsed (41 bits for DS and 64 bits for APFS and NTFS), we can determine the earliest and the latest time point defined of the time domain. Since DS and NTFS do not allow backward counting, DS can theoretically store elapsed times in the range from 0 to $2^{41} - 1 = 2,199,023,255,551$ milliseconds and, NTFS from 0 to $2^{64} - 1 = 18,446,744,073,709,551,615$ hundreds of nanoseconds. In contrast, APFS allows value from the range from $-2^{63} = -9,223,372,036,854,775,808$ to $2^{63} - 1 = 9,223,372,036,854,775,807$ nanoseconds. Picking the epoch as the time point corresponding to the value 0 of the valid range, we can then compute the earliest and latest time point defined. Without taking leap seconds into account due to their unpredictability, their time domains, denoted by D_{APFS} , D_{NTFS} , and D_{DS} , are defined as follows.

- D_{APFS} : Every nanosecond between September 21, 1677 00:12:43.145224192 UTC and April 11, 2262 23:47:16.854775807 UTC excluding leap seconds
- D_{NTFS} : Every 100 nanoseconds between January 1, 1601 00:00:00.0000000 UTC and 28 May, 60056 05:36:10.9551615 excluding leap seconds
- D_{DS} : Every millisecond between January 1, 2015 00:00:00.000 UTC and September 6, 2084 15:47:35.551 UTC excluding leap seconds

2.2.3 Forensic Analysis of Timestamps

Digital investigations require to interpret obtained traces, including timestamps, from different sources. For instance, a case might require to analyze a USB-stick with exFAT, a phone with APFS and a laptop with NTFS. To enable consistent handling, comparison, and

searching across these sources, one needs to establish links between timestamps across distinct underlying time domains. Drawing these links refers to the task of *timestamp normalization*, informally defined as follows.

► **Definition 1.** *Timestamp normalization refers to the process of representing timestamps from different time domains in a single time domain.*

These normalizations cannot be performed arbitrarily, but need to adhere to principle of *forensic soundness* which is essential to obtain admissible evidence for criminal proceedings. This concept refers to the ideals of *integrity* and *chain of custody*. Integrity demands that the preservation of evidence and its meaning be maintained. However, this does not imply that digital evidence has to be preserved in its entire form, as acquiring data can already lead to modification of data. This is why the ideal of chain of custody demands that any action performed on digital evidence is documented [4, 11, 40]. For the normalization of timestamps, we recognize the following implications of forensic soundness:

- To ensure the integrity of timestamps, the temporal relationships within the same domain must be preserved. However, integrity does not require that a time point may only be normalized with the same time point. As we will discuss later, this flexibility is necessary to normalize time points with different resolutions.
- In light of the ideal of chain of custody, any normalization procedure and the resulting links between timestamps across distinct domains must be grounded in a definition to guarantee explainability, accounting for these potential differences in representation of time points.

The second aspect emphasizes the importance of a formal definition of timestamp normalization for ensuring compliance with these requirements.

3 A Formal Framework for Timestamp Normalizations

In this section, we formalize the notion of forensically sound timestamp normalization. To do this, we first define normalizations of time domains (Section 3.3), which builds on core notions like time domains and their intervals (Sections 3.1 and 3.2). We then formally define timestamp normalization (Section 3.6) and reveal a critical insight: normalized time points may not be comparable, which is related to the resolution of the time domain (Sections 3.4 and 3.5) and affects the comparability of normalized timestamps.

3.1 Introducing the Core Notions

Time Domain. As discussed informally in Section 2, time domains are sets of time points which refer to specific moments in time. We denote time domains by capital letters like D and T . Formally, any time domain D is a non-empty (possibly countably infinite) set of time points. The time domains D_{exFAT}^1 , D_{exFAT}^2 , D_{APFS} , D_{NTFS} , and D_{DS} described in the previous section are examples of time domains. We denote any time point of some time domain by t . Examples of time points are t_0 and t_1 .

$$t_0 = \text{January 1, 2020 00:00:00 UTC} \quad t_1 = \text{January 1, 2020 00:00:00.000 UTC}$$

Importantly, t_0 and t_1 do *not* refer to the same moment in time because the latter contains more temporal information than the former one due to the different resolution.

Time Domain Relation. We assume that for any time domain in digital systems, its time points are comparable. Given any t_1 and t_2 of the same time domain, one knows whether t_1 follows t_2 or vice versa. This clearly holds for D_{exFAT}^1 , D_{exFAT}^2 , D_{APFS} , D_{NTFS} , and D_{DS} .

Formally, for any time domain D , we fix a time domain relation $\preceq_D$ on D so that $\preceq_D$ is a total order. That is, $\preceq_D \subseteq D \times D$ is reflexive, antisymmetric, transitive and total. A time domain relation expresses the temporal order within the domain considered.

We fix that for any time domains D and T , if $t_1, t_2 \in D \cap T$, then $t_1 \preceq_D t_2$ if and only if $t_1 \preceq_T t_2$. As a consequence, if $D = T$, then $\preceq_D = \preceq_T$. This condition ensures that any time points shared between different domains preserve the same ordering across all domains.

We further introduce the follow notation: $t_1 \not\preceq_D t_2$ denotes that it does *not* hold that $t_1 \preceq_D t_2$, while $t_1 \prec_D t_2$ denotes that $t_1 \preceq_D t_2$ and $t_2 \not\preceq_D t_1$. Further, $t_1 =_D t_2$ denotes that $t_1 \preceq_D t_2$ and $t_2 \preceq_D t_1$. If the context is clear, we omit the reference to the respective time domain D for $\preceq_D$, $\prec_D$, and $=_D$. That is, $\preceq$, $\prec$, and $=$ denote the time domain relation, its strict part and its equivalence part of D .

Timestamps. Digital system do not provide timestamps for arbitrary events, but for a specific list of events. For example, a file system timestamps the creation and last access time of files, among other things. An event set provides such events and we denote them by E . Formally, an event set is a non-empty (possibly countably infinite) set and we denote an event of E by e . The information that an event occurred is stored as an artifact in form of a timestamp. Given a set of events E and a time domain D , a timestamp relative to E and D is a tuple $\langle e, t \rangle$ with $e \in E$ and $t \in D$. A timestamp is therefore a pair consisting of a trace and a time point specifying one time point of D where e was registered.

Given two timestamps, $\langle e_1, t_1 \rangle$ and $\langle e_2, t_2 \rangle$, relative to the same time domain $t_1, t_2 \in D$, we can then express that event e_1 tracked by $\langle e_1, t_1 \rangle$ (*weakly*) *precedes* e_2 tracked by $\langle e_2, t_2 \rangle$ using the time domain order $\preceq_D$ on D . That is, e_1 tracked by $\langle e_1, t_1 \rangle$ precedes e_2 tracked by $\langle e_2, t_2 \rangle$ if $t_1 \preceq_D t_2$.

3.2 Intervals of Time Domains

As we will discuss in Section 3.3, normalizations of time domains require intervals of time points. Intuitively, an interval of a time domain is a set of time points that lie between an upper and a lower bound.

► **Definition 2.** Let D be a time domain and $\preceq$ be its time domain relation. For any $t_1, t_2 \in D$, $[t_1, t_2]$ is defined as $[t_1, t_2] = \{t \in D \mid t_1 \preceq t \preceq t_2\}$. The set $\mathcal{I}_D^{\preceq}$ of (non-empty) intervals of D relative $\preceq$ is defined as $\mathcal{I}_D^{\preceq} = \{[t_1, t_2] \mid t_1, t_2 \in D \wedge t_1 \preceq t_2\}$.

We say that $[t_1, t_2]$ is an interval of D and $\preceq$, if $[t_1, t_2] \in \mathcal{I}_D^{\preceq}$. For any time domain D and any $t \in D$, $[t, t] = \{t\}$ is an interval of D , because any time domain relation $\preceq$ is reflexive. We refer to $[t, t]$ as the *time point interval* of t . Further, the definition of the set of (non-empty) intervals enforces that no $\mathcal{I}_D^{\preceq}$ contains any empty sets or is empty.

► **Proposition 3.** For any time domain D and its time domain relation $\preceq$, (i) $\mathcal{I}_D^{\preceq} \neq \emptyset$, and (ii) for any $[t_i, t_j] \in \mathcal{I}_D^{\preceq}$ it holds that $[t_i, t_j] \neq \emptyset$.

Building on the fixed time domain relation of a time domain, we define a relation on the set of time intervals of that domain as follows.

► **Definition 4.** Given a time domain D and its time domain relation $\preceq$, the interval relation $\preceq_D^{\mathcal{I}} \subseteq \mathcal{I}_D^{\preceq} \times \mathcal{I}_D^{\preceq}$ is defined as for any $[t_1, t_2], [t_3, t_4] \in \mathcal{I}_D^{\preceq}$, $[t_1, t_2] \preceq_D^{\mathcal{I}} [t_3, t_4] \iff t_2 \preceq t_3$.

Since $\preceq$ is reflexive, the interval relation $\preceq_D^{\mathcal{I}}$ allows ordered intervals to overlap. That is, if $t_2 = t_3$, then $[t_1, t_2] \preceq_D^{\mathcal{I}} [t_3, t_4]$. This is why, we say that $[t_1, t_2]$ *weakly precedes* $[t_3, t_4]$.³ Further, unlike the underlying time domain relation, the interval relation is neither reflexive nor total if the time domain contains more than one time point.

► **Proposition 5.** *Let D be any time domain, $\preceq$ be its time domain relation, and $\preceq_D^{\mathcal{I}} \subseteq \mathcal{I}_D^{\preceq} \times \mathcal{I}_D^{\preceq}$ be its interval relation.*

- (1) $\preceq_D^{\mathcal{I}}$ is transitive and antisymmetric.
- (2) If $|D| = 1$, then $\preceq_D^{\mathcal{I}}$ is reflexive and total.
- (3) If $|D| > 1$, then $\preceq_D^{\mathcal{I}}$ is neither reflexive nor total.

3.3 Time Domain Normalization

As informally specified in Definition 1, normalizing a time domain requires characterizing its time points in terms of time points from another time domain. We formally define normalization of a time domain as follows.

► **Definition 6.** *Let D and T be time domains, and $\preceq_D$ and $\preceq_T$ be their time domain relations. A normalization of D relative to T is a partial function $n : D \rightarrow \mathcal{I}_T^{\preceq}$ such that*

- (1) *if $n(t_1)$ and $n(t_2)$ are defined for $t_1, t_2 \in D$, then $n(t)$ is defined for all $t \in D$ with $t_1 \preceq_D t \preceq_D t_2$;*
- (2) *for any $t_1, t_2 \in D$ with $n(t_1)$ and $n(t_2)$ defined, if $t_1 \prec_D t_2$, then $n(t_1) \preceq_T^{\mathcal{I}} n(t_2)$;*
- (3) *for any $t \in D \cap T$, $n(t) = [t, t]$.*

For any $t \in D$, if $n(t)$ is defined, we call $n(t)$ the normalization of t relative to n and T .

Accordingly, a normalization is a partial function whose domain is a time domain (i.e., D) and whose codomain is a set of intervals (i.e., $\mathcal{I}_T^{\preceq}$). A set of intervals as the codomain is necessary to accommodate time domains of different resolution. Consider, for instance, D_{DS} and D_{APFS} . A time point t in D_{DS} with millisecond precision corresponds to 1,000,000 time points in D_{APFS} , each representing a time point with nanosecond precision. Due to the hierarchical composition of tracked time, t is not equivalent to any single one of these 1,000,000 time points in D_{APFS} , but to the set of all of them.

We can express the fact that not all time points in the domain are necessarily normalized by looking at the partiality of normalizations. This property provides sufficient flexibility when the time domain to be normalized and the time domain used for normalization do not cover the same period of time. For example, D_{exFAT}^1 contains time points starting from December 31, 1979 UTC, whereas the earliest defined time points in D_{DS} begin on January 1, 2015 UTC. If normalizations were not partial, attempting to normalize D_{exFAT}^1 relative to D_{DS} would require assigning time points from the 20th century to intervals occurring at least 35 years later. The forensic soundness would be violated by such a drastic redating.

Constraint (1) limits where a normalization may be undefined. It requires the set of time points for which the normalization is defined must form a contiguous segment of the time domain. As a consequence, the normalization of time points at the boundaries can be undefined (assuming that the normalization of some time points is defined). In the previous example of normalizing D_{exFAT}^1 relative to D_{DS} , this allows all time points before

³ Analogously, strict precedence on intervals is definable by demanding $t_2 \prec t_3$. However, only weak precedence is required to express order preservation during normalization across different resolutions. Nevertheless, future research should address strict precedence when constructing timelines of events based on normalized timestamps.

January 1, 2015 UTC and after September 6, 2084 UTC to remain undefined, while requiring the normalization to be defined for all intermediate time points of D_{exFAT}^1 , if time points of D_{exFAT}^1 are normalized by their counterparts in D_{DS} .

Constraint (2) demands that a normalization does not alter the ordering of time points during a normalization. To ensure this property, if for $t_1, t_2 \in D$ it holds that $t_1 \prec_D t_2$, then for their normalized values $n(t_1)$ and $n(t_2)$, it must hold that interval $n(t_1)$ weakly precedes $n(t_2)$. Note that the inverse does not hold, as illustrated in Example 7.

► **Example 7.** Let us consider the normalization of $D = D_{\text{exFAT}}^1$ relative to $T = D_{\text{exFAT}}^2$ by focusing on the following time points $t_1, \dots, t_{199} \in D$ and $t \in T$.

$$\begin{aligned} t_0 &= \text{January 1, 2000 00:00:00.00 UTC} & t &= \text{January 1, 2000 00:00:00 UTC} \\ &\vdots & & \\ t_{199} &= \text{January 1, 2000 00:00:01.99 UTC} \end{aligned}$$

We define the normalization n of D relative to T as $n(t_0) = \dots = n(t_{199}) = [t, t]$. Since D_{exFAT}^1 is hence normalized by a time domain with a more coarse resolution, time points distinguishable by the time domain relation of D like t_0 and t_{199} (i.e., $t_0 \prec_D t_{199}$), are not distinguishable after the normalization because $n(t_0) = n(t_{199})$. Further, because they are both normalized by the point interval of t , $n(t_{199}) \preceq_T^I n(t_0)$, while $t_{199} \not\prec_D t_0$.

Lastly, let us discuss constraint (3) that considers normalizations where the time domain to be normalized contains time points also contained by the time domain relative to which the former is normalized; i.e., $D \cap T \neq \emptyset$. For the time points occurring in both time domains, it is required that they are normalized by their time point interval. For these values, the interval relation and the time domain relation encode the same ordering.

► **Proposition 8.** Let D and T be time domains, $\preceq_D$ and $\preceq_T$ be their time domain order, and $n : D \rightarrow \mathcal{I}_T^{\preceq_T}$ be any normalization of D relative to T . If $D \cap T \neq \emptyset$, then for any $t_1, t_2 \in D \cap T$, $t_1 \preceq_D t_2 \iff n(t_1) \preceq_T^I n(t_2)$.

Besides these constraints, one can define further properties of normalizations.

► **Definition 9.** Let D and T be time domains, and $\preceq_D$ and $\preceq_T$ be their time domain relations. A normalization $n : D \rightarrow \mathcal{I}_T^{\preceq_T}$ is

- (1) *non-empty* if $n(t)$ is defined for some $t \in D$;
- (2) *gap-free* if for any $t_1, t_2 \in T$ with $t_1 \in n(t'_1)$ and $t_2 \in n(t'_2)$ for some $t'_1, t'_2 \in D$, then for any t with $t_1 \preceq_T t \preceq_T t_2$, there exist $t' \in D$ such that $t \in n(t')$;
- (3) *even* if for any $t_1, t_2 \in D$ with $n(t_1)$ and $n(t_2)$ defined, $|n(t_1)| = |n(t_2)|$.

Non-emptiness indicates that, under the given normalization, the time domains overlap, ensuring that at least one point of D is normalized by an interval of T . Gap-freeness and evenness guarantee that the normalized time points (and the intervals that normalize them) are uniformly distributed over the target time domain. This uniformity is crucial when the domain itself is evenly spaced, such as all time domains based on UTC.

3.4 Comparing Time Domain Resolutions relative to Normalizations

Normalizations enable us to compare the resolution of time domains. That is, given a normalization of a time domain relative to another time domain, we can evaluate whether the former is more coarse or fine-grained than the latter. As we will discuss in Section 3.5, differences in resolution affect the comparability of normalized time points originating from different time domains.

► **Definition 10.** Let D and T be time domains and let $n : D \rightarrow \mathcal{I}_T^{\leq T}$ be a normalization of D relative to T with $\text{Dom}(n) = \{t \in D \mid n(t) \text{ defined}\}$. We say that

- D is at $t \in \text{Dom}(n)$ strictly more fine-grained than T relative to n (denoted $T \triangleleft_n^t D$), if there exist $t' \in D$ such that $t \neq t'$ and $n(t) = n(t')$. Further, D is globally strictly more fine-grained than T relative to n (denoted $T \triangleleft_n^\top D$), if for any $t \in \text{Dom}(n)$, $T \triangleleft_n^t D$.
- D is at $t \in \text{Dom}(n)$ at least as fine-grained as T relative to n (denoted $T \preceq_n^t D$), if $|n(t)| = 1$. Further, D is globally at least as fine-grained as T relative to n (denoted $T \preceq_n^\top D$), if for any $t \in \text{Dom}(n)$, $T \preceq_n^t D$.
- D is at $t \in \text{Dom}(n)$ strictly less fine-grained than T relative to n (denoted $D \triangleleft_n^t T$), if $|n(t)| > 1$. Further, D is globally strictly less fine-grained than T relative to n (denoted $D \triangleleft_n^\top T$), if for any $t \in \text{Dom}(n)$, $D \triangleleft_n^t T$.

Strictly-more-fine-grainedness requires comparing the normalization of multiple time points. Intuitively, a time domain is strictly more fine-grained than another when multiple distinct points in the former normalize to the same value in the latter. For example, seconds are strictly more fine-grained than minutes because 60 distinct seconds normalize to the same minute. At-least-as-fine-grainedness does not demand the comparison of two time-points. In Proposition 11, it will be shown that strictly-more-fine-grainedness implies at-least-as-fine-grainedness. Strictly-less-fine-grainedness occurs when a time point in the domain corresponds to multiple normalized values. For instance, an hour is strictly less fine-grained than minutes because one hour normalizes to 60 minutes. These distinctions lead to the definitions above. Independent of the different definitions, these notions intuitively relate to each other. For instance, if a domain is locally (i.e., at a time point) or globally strictly more fine-grained than the domain used to normalize, it implies that it is also locally or globally at least as fine-grained as the other domain, but not locally or globally strictly less fine-grained. This intuitive property, among others, is given by Proposition 11.

► **Proposition 11.** Let D and T be time domains, while $n : D \rightarrow \mathcal{I}_T^{\leq T}$ be a non-empty normalization of D relative to T with $\text{Dom}(n) = \{t \in D \mid n(t) \text{ defined}\}$. For any $t \in \text{Dom}(n)$,

- (1) $D \triangleleft_n^t T$ if and only if $T \not\preceq_n^t D$;
- (2) if $T \triangleleft_n^t D$ then $T \preceq_n^t D$;
- (3) if $T \triangleleft_n^t D$ then $D \not\triangleleft_n^t T$;
- (4) if $D \triangleleft_n^\top T$ then $T \not\preceq_n^\top D$;
- (5) if $T \triangleleft_n^\top D$ then $T \preceq_n^\top D$;
- (6) if $T \triangleleft_n^\top D$ then $D \not\triangleleft_n^\top T$.

3.5 Comparing Normalized Time Domains

Normalizing multiple time domains relative to the same time domain provides the foundation to compare these time points. However, normalization relative to the same time domain does not ensure that the normalized time points of the distinct time domains are comparable. This means that we are not able to state whether the one normalized time point happened before or after the other normalized time point (in the sense of weak precedence). Such incomparability is formally due to the fact that the interval time domain order $\preceq_D^{\mathcal{I}}$ is not total for a time domain D containing more than one time point, as illustrated by Proposition 5.

► **Example 12.** As illustrated in Section 2.2, exFAT file system provides last access timestamps with a 2 seconds resolution, while NTFS provides file timestamps, including last access timestamps, with a resolution of 100 nanoseconds. An example of time points of these domains are $t \in D_{\text{NTFS}}$ and $t' \in D_{\text{exFAT}}^2$ which are defined as follows.

$$t = \text{January 1, 2026 10:00:00.5000000 UTC} \quad t' = \text{January 1, 2026 10:00:00 UTC}$$

10:10 A Formal Model for Forensically Sound Timestamp Normalization

If t and t' are normalized relative to a time domain T of resolution milliseconds (e.g., $T = D_{\text{DS}}$), it is impossible to determine which time point was registered first. This is due to the coarser resolution of D_{exFAT}^2 . To make this circumstance explicit, we normalize both time points relative to D_{DS} . We define two normalizations n_1 and n_2 such that $n_1(t) = [t_{500}, t_{500}]$ and $n_2(t') = [t_0, t_{1999}]$ where $t_0, \dots, t_{1999}$ are defined as follows.

$$\begin{array}{ll} t_0 = \text{January 1, 2026 10:00:00.000 UTC} & t_{1000} = \text{January 1, 2026 10:00:01.000 UTC} \\ \vdots & \vdots \\ t_{500} = \text{January 1, 2026 10:00:00.500 UTC} & t_{1500} = \text{January 1, 2026 10:00:01.500 UTC} \\ \vdots & \vdots \\ t_{999} = \text{January 1, 2026 10:00:00.999 UTC} & t_{1999} = \text{January 1, 2026 10:00:01.999 UTC} \end{array}$$

The fact that $t_0 \prec_T t_{500} \prec_T t_{1999}$ highlights that we have no information about the relative temporal ordering of these normalized time points.

Generally speaking, incomparable normalized time points occur if the intervals constituting their normalizations overlap so that, analogously to the discussed example, one is not capable to state which normalized time point weakly precedes the other normalized time point.

► **Proposition 13.** *Let D_1 , D_2 , and T be time domains, and n_1 and n_2 be normalizations of D_1 and D_2 relative to T . For any $t \in D_1$ and $t' \in D_2$ with $n_1(t) = [t_1, t_2]$ and $n_2(t') = [t_3, t_4]$, $n_1(t) \not\preceq_T^{\mathcal{I}} n_2(t')$ and $n_2(t') \not\preceq_T^{\mathcal{I}} n_1(t)$ if and only if $t_3 \prec_T t_2$ and $t_1 \prec_T t_4$.*

Incomparability of distinct time points occurs only when normalized time points from different domains are considered and these time domains, including the domain used for the normalizations, are not identical. This result is captured by the following proposition.

► **Proposition 14.** *Given the time domains D_1 , D_2 , and T such that $D_1 = D_2 = T$, then for any normalizations n_1 and n_2 of D_1 and D_2 relative to T respectively, $n_1(t_1) \preceq_T^{\mathcal{I}} n_2(t_2)$ or $n_2(t_2) \preceq_T^{\mathcal{I}} n_1(t_1)$ for any $t_1 \in D_1$ and $t_2 \in D_2$.*

As previously mentioned, the (in)comparability of normalized time points from different domains is related to the differences in resolution between the normalized time domains and the domain used for normalization. Proposition 15 illustrates one aspect of this connection.

► **Proposition 15.** *Let D_1 , D_2 , and T be time domains, and n_1 and n_2 be normalization of D_1 and D_2 relative to T , respectively.*

- (1) *If $T \trianglelefteq_{n_1}^{\top} D_1$ and $T \trianglelefteq_{n_2}^{\top} D_2$, then $n_1(t_1) \preceq_T^{\mathcal{I}} n_2(t_2)$ or $n_2(t_2) \preceq_T^{\mathcal{I}} n_1(t_1)$ for any $t_1 \in D_1$ and $t_2 \in D_2$ with $n_1(t_1)$ and $n_2(t_2)$ defined.*
- (2) *If $T \trianglelefteq_{n_1}^{\top} D_1$ and $D_2 \triangleleft_{n_2}^{\top} T$ where $|n_2(t)| = 2$ for any $t \in \{t \in D_2 \mid n_2(t) \text{ defined}\}$, then $n_1(t_1) \preceq_T^{\mathcal{I}} n_2(t_2)$ or $n_2(t_2) \preceq_T^{\mathcal{I}} n_1(t_1)$ for any $t_1 \in D_1$ and $t_2 \in D_2$ with $n_1(t_1)$ and $n_2(t_2)$ defined.*

Note that (1) also holds if D_1 or D_2 are globally strictly more fine-grained than T , because, as stated in Proposition 11, strictly-more-fine-grainedness implies at-least-as-fine-grainedness. Analogously, (2) also holds if D_1 is globally strictly more fine-grained than T .

Further, proposition (1) suggests that a less fine-grained time domain used for normalization can ensure comparability between time points from different time domains. However, such a less fine-grained time domain implies a loss of resolution in distinguishing time points which means that information is lost. Proposition (2) sheds lights on a relevant fact for

the digital forensic domain. Although digital systems like exFAT use time domains with a resolution of two seconds, normalizing it to a time domain with second resolution does not lead to incomparable time points if another time domain with finer-grained resolution is normalized to the same time domain.

Nevertheless, comparability of time points of different time domains after normalization relative to one time domain cannot be guaranteed. Proposition 16 investigates some of these situations that are forensically interesting. It articulates the inevitability of incomparable normalized time points when two time domains are normalized relative to a reference time domain that is at least as fine-grained as at least one of the time domains being normalized, and when the overlap of the normalized time points contains a sufficiently large number of time points. A natural example is the normalization of time domains, such as D_{DS} and D_{AFPS} , with millisecond and nanosecond resolution, respectively, relative to a time domain with microsecond resolution. Clearly, the overlap of the normalized values should exceed the required three microseconds.

► **Proposition 16.** *Let D_1 , D_2 and T be time domains, and n_1 and n_2 be gap-free normalizations of D_1 and D_2 relative to T , respectively.*

- (1) *If $T \triangleleft_{n_1}^\top D_1$, $D_2 \triangleleft_{n_2}^\top T$ where $|n_2(t)| \geq 3$ for any $t \in \{t \in D_2 \mid n_2(t) \text{ defined}\}$, and $|\{t \in T \mid t \in n_1(t'_1) \text{ for some } t'_1 \in D_1\} \cap \{t \in T \mid t \in n_2(t'_2) \text{ for some } t'_2 \in D_2\}| \geq 3$, then there exist $t_1 \in D_1$ and $t_2 \in D_2$ such that $n_1(t_1) \not\preceq_T^\mathcal{I} n_2(t_2)$ and $n_2(t_2) \not\preceq_T^\mathcal{I} n_1(t_1)$.*
- (2) *If $D_1 \triangleleft_{n_1}^\top T$ where $|n_1(t)| \geq 3$ for any $t \in \{t \in D_1 \mid n_1(t) \text{ defined}\}$, $D_2 \triangleleft_{n_2}^\top T$ where $|n_2(t)| \geq 3$ for any $t \in \{t \in D_2 \mid n_2(t) \text{ defined}\}$, and $|\{t \in T \mid t \in n_1(t'_1) \text{ for some } t'_1 \in D_1\} \cap \{t \in T \mid t \in n_2(t'_2) \text{ for some } t'_2 \in D_2\}| \geq 2$, then there exist $t_1 \in D_1$ and $t_2 \in D_2$ such that $n_1(t_1) \not\preceq_T^\mathcal{I} n_2(t_2)$ and $n_2(t_2) \not\preceq_T^\mathcal{I} n_1(t_1)$.*

3.6 Timestamp Normalization

In the next step, we want to normalize timestamps. Recall that, given a set of events E and a time domain D , a timestamp relative to E and D is tuple $\langle e, t \rangle$ with $e \in E$ and $t \in D$ denoting that e was registered at t . To normalize this timestamp relative to time domain, a normalization of the time domain underlying the timestamp relative to this other time domain is needed. If such a normalization is given, we normalize the timestamp by normalizing its time point. We define this process as follows.

► **Definition 17.** *Let D and T be time domains, E be a set of events, and n be a normalization of D relative to T . Any timestamp $\langle e, t \rangle$ with $e \in E$ and $t \in D$ is normalizable by n relative to T if $n(t)$ is defined. For any normalizable timestamp $\langle e, t \rangle$ with $e \in E$ and $t \in D$, we define the the normalization of $\langle e, t \rangle$ relative to n and T , denoted $n(\langle e, t \rangle)$, as $\langle e, n(t) \rangle$.*

As a consequence of this definition, timestamps and normalized timestamps conceptually differ. Timestamps are pairs containing one event and one time point, while normalized timestamps are pairs containing one event and a set of time points (i.e., the interval that normalizes the time point). This conceptual difference is an inescapable consequences of accounting for different time domains in the process of normalizations.

Timestamp normalization allows us to infer the order of events which are tracked using different time domains. Given two timestamps $\langle e_1, t_1 \rangle$ and $\langle e_2, t_2 \rangle$ based on time domains D_1 and D_2 , and their normalizations $n_1(\langle e_1, t_1 \rangle)$ and $n_2(\langle e_2, t_2 \rangle)$ relative to T , we can establish an order between the tracked events e_1 and e_2 using the interval time domain order $\preceq_T^\mathcal{I}$. That is, e_1 tracked by normalized timestamp $n_1(\langle e_1, t_1 \rangle)$ weakly precedes e_2 tracked by normalized timestamp $n_2(\langle e_2, t_2 \rangle)$ if $n_1(t_1) \preceq_T^\mathcal{I} n_2(t_2)$ and vice versa.

The findings discussed in Section 3.5 regarding incomparable normalized time points affect the comparability of normalized timestamps. If the underlying time domains of two timestamps contain incomparable normalized time points (as, for instance, discussed in Proposition 16), the timestamps may become incomparable. This means the order of events cannot be determined. This occurs when for $n_1(\langle e_1, t_1 \rangle)$ and $n_2(\langle e_2, t_2 \rangle)$ it holds that $n_1(t_1) \not\leq_T^{\mathcal{I}} n_2(t_2)$ and $n_2(t_2) \not\leq_T^{\mathcal{I}} n_1(t_1)$. For a forensically sound analysis, it must therefore always be taken into account that timestamps, despite an underlying time domain where every point in time is comparable, become incomparable through normalization. In Section 4, we will see that state-of-the-art forensic software does not fulfill this demand.

4 Normalization in Forensic Software

In this section, we evaluate the timestamp normalization implementations in Plaso version 20260119, Sleuth Kit version 4.14.0., and Hansken version 47.42.0. We show how they violate our proposed model of forensically sound normalization, illustrating the negative impact with a real-world example.

4.1 Timestamp Normalization in Plaso, Sleuth Kit, and Hansken

Plaso. Plaso [35] represents timestamps from various sources on a unified timeline. For this purpose, it parses artifacts from multiple sources, such as file systems, logs, and cache data. Given this parsed information, it represents the timestamp events on one timeline. Plaso normalizes timestamps into a standardized string representation of the form `YYYY-MM-DD hh:mm:ss.#####[±]HH:MM`. Only the date component (`YYYY-MM-DD`) is mandatory, while the time of day (`hh:mm:ss`), fractional seconds with milli- or microsecond resolution (`#####`), and timezone offset with minute resolution relative to UTC (`[±]HH:MM`) are optional. If no timezone is specified, UTC is assumed.⁴

Sleuth Kit. The Sleuth Kit [36] is frequently used in forensic analysis for volume and file system analysis. The aim of this analysis is to make metadata of files within a file system, like the timestamps of a file, accessible for forensic investigations. Sleuth Kit normalizes file timestamps by converting any timestamp in a UNIX timestamp. For the considered file systems, any exFAT timestamps is converted into the a single positive number of seconds since January 1, 1970 00:00:00 UTC, while any NTFS or APFS timestamp is converted into a single positive number of nanoseconds since this epoch.

Hansken. Hansken [33, 41] extracts traces from digital data, systemizes them, and provides tools to efficiently search this data. It extracts timestamps from various sources and normalizes them into a standardized string of the form `YYYY-MM-DD hh:mm:ss.sss Z`, where `Z` is the standard notation for UTC timezone. The date (`YYYY-MM-DD`) and time (`hh:mm:ss.sss Z`) are both mandatory and are always given relative to UTC. Further, Hansken provides the resolution of the original timestamp, but not the timestamp itself.

⁴ Assigning a default timezone to timestamps lacking timezone information refers to the process of inferring missing temporal context. We briefly outline this challenge as future work in Section 6.

4.2 Assessing Alignment with the Proposed Model

Unlike our model of forensically sound timestamp normalization (Sections 3.3 and 3.6), Plaso, Sleuth Kit, and Hansken normalize timestamps by converting time points into a different time domain, which may have a different resolution. Consequently, their method does not align with ours as it does not utilize intervals. This discrepancy leads to misleading normalized timestamps, as the following example illustrates.

► **Example 18.** Using a script, we created multiple `.txt` files on exFAT and NTFS file systems on Windows 11. Filenames include a sequential creation index and a timestamp with millisecond resolution (UTC+02:00) to track the order and time of file creation. ExFAT and NTFS store a file’s creation time and last access time in its metadata. Since no file was accessed after creation, these timestamps refer to the same time in the corresponding time domain. As discussed in Section 2.2, exFAT tracks the former with a resolution of 10 milliseconds and the latter with a resolution of 2 seconds, while NTFS timestamps both with a resolution of 100 nanoseconds. The following table shows the extracted normalized timestamps of two created files.

■ **Table 1** Timestamps extracted and normalized by Plaso, Sleuth Kit and Hansken for two files.

Software	Create Timestamp	Last Access Timestamp
File 02.2026-04-10T09-07-04.484.txt stored in NTFS file system:		
Plaso	2026-04-10T07:07:04.484325+00:00	2026-04-10T07:07:04.484325+00:00
Sleuth Kit	2026-04-10 09:07:04.484325000 (CEST)	2026-04-10 09:07:04.484325000 (CEST)
Hansken	2026-04-10T07:07:04.484Z	2026-04-10T07:07:04.484Z
File 05.2026-04-10T09-07-05.992.txt stored in exFAT file system:		
Plaso ^{a)}	2026-04-10T09:07:05.000000+00:00	2026-04-10T09:07:04.000000+00:00
Sleuth Kit	2026-04-10 09:07:05 (CEST)	2026-04-10 09:07:04 (CEST)
Hansken	2026-04-10T07:07:05.990Z	2026-04-10T07:07:04.000Z

a) Plaso incorrectly uses UTC instead of UTC+02:00 for both timestamps.

None of the software informs users when a timestamp has been normalized relative to a domain with a different resolution than the original. However, due to the resolution of the original timestamps provided, Hansken provides clues to such differences. Further, none of the tools provide information indicating that the time points given as the normalized value should be interpreted as intervals. Consequently, the timestamps appear to be time points, as shown in Table 1. This can lead to the following undesired conclusions.

Although the create and the last access timestamp within exFAT track the same moment in time when a file is created, the normalized timestamps of the forensic tools incorrectly suggest that 02...txt was created after it was last accessed. Furthermore, while 02...txt was created and hence accessed prior to 05...txt, a comparison limited to 02...txt’s last access time and 05...txt’s creation time reveals a temporal inversion. The normalized last access time of 02...txt appears after the normalized creation time of 05...txt. This data incorrectly implies that 05...txt was accessed before 02...txt was created. In a forensic investigation aimed at reconstructing the sequence of events, such discrepancies could lead to incorrect conclusions regarding the chronology of created and accessed files.

Importantly, forensically unsound conclusions based on timestamps normalized by these tools can be avoided by representing normalized timestamps as time intervals rather than single points. For example, normalizing 02...txt’s create timestamp and 05...txt’s last

access timestamp with a time interval (e.g., a single point interval of 2026-04-10 09:07:04.484 CEST and a range interval from 2026-04-10 09:07:04.000 CEST to 2026-04-10 09:07:06.000 CEST, respectively) would lead to the forensically sound conclusion that one cannot determine whether `05...txt` was accessed before or after `02...txt`.

5 Related Work

To our knowledge, timestamp normalization in digital forensics has not been formally studied as a distinct problem, yet our work relates to research in forensic science, temporal logic including models of different temporal granularities, and temporal expression processing.

Within forensic science, connecting artifacts across domains has received some attention. One approach [19] connects time domains, including ones underlying implicit and explicit timing information (e.g., file IDs and file timestamps), but focuses on making implicit timing information available for forensic investigations rather than normalizing timestamps. Another [12] provides a formal framework for reasoning about timelines from multiple trace sources, but presupposes normalized timestamps. Our work establishes a formal foundation for performing timestamp normalization in a forensically sound manner.

Our proposed representation of tracked time, specified by time domains, time domain relations and intervals, is based on instant-based and interval-based formal models of time [1, 21, 42], which have been applied in the legal domain to, for instance, encode transportation regulations [17]. The time domain normalization and resolution definitions (Definitions 6 and 10) relate to time-domain based models of temporal granularity⁵. Prior work, surveyed in [16, 20], proposes the following domain-based approaches. Approach (i) [15] defines a sequence of time domains, ordered from finest to coarsest granularity, where each domain (except the first) maps to a finite interval of the previous domain, covering all its elements. Approach (ii) [14, 31] relaxes this constraint, modeling levels of granularity as disjoint time domains and stipulating a total order on them to express granularity comparisons. Both (i) and (ii) define a mapping from time points of a coarser domain to intervals of time points in a more fine-grained domain, while (ii) includes a reverse mapping from time points of a more fine-grained domain to a single time point in a more coarse domain. Our approach to time domain normalization relates to these mappings (like them, it maps time points to intervals), but it and our notion of granularity (i.e., resolution) differ. In our approach, non-normalized (i.e., unmapped) time points are permitted; during normalization, time points are always mapped to intervals; and differences in granularity are defined based on normalizations. Approach (iii) [6, 7] defines levels of granularity as mappings from an infinite time domain $\mathbb{R}$ to sets of non-overlapping intervals on $\mathbb{R}$, deriving fine-grainedness comparisons from set-theoretic relations and providing interval-to-interval mappings to relate temporal information across different levels of granularity. Normalizing multiple time domains relative to the same time domain results in distinct sets of intervals on the latter that resemble levels of granularity according to (iii). However, their mappings and granularity relations diverge from ours, as they allow reasoning about levels of granularity (i.e., resolution) after the normalization (interval-to-interval level) and not during the normalization (time-point-to-interval level).

Lastly, handling temporal data is not unique to digital forensics. It is addressed in many fields, including natural language processing [13, 39], medicine [27], archaeology [8], ontology languages [2], and operating systems [22]. They aim to provide a uniform format

⁵ In the literature on formal models of temporal granularity, the notion of “granularity”, similar to the notion of “resolution” used in this paper, refers to different units of time (e.g., seconds, minutes, hours).

for dates and temporal expressions, such as ISO 8601 [24]. Our work shares this goal, but differs in scope. We provide a model of forensically sound timestamp normalization across heterogeneous digital systems rather than parsing temporal expressions and dates.

6 Conclusion and Discussion

The analysis of digital devices such as smartphones and laptops is currently indispensable in criminal investigations. Given the critical role of digital investigations, these investigations also carry a significant responsibility to uphold justice, accuracy, and accountability. In order to live up to these values, digital forensic processes must align with the principles of forensic soundness. In particular, it is crucial to ensure that automated forensic software processes are well understood and adhere to these principles.

Summary. This paper contributes to addressing this fundamental challenge by providing and investigating a formalization of forensically sound timestamp normalization (Sections 3.3 and 3.6), which is a key process that enables cross-device and cross-system timestamp analysis. This formalization not only serves as a novel foundation for timestamp normalization and thus a potential basis for future implementations, but also reveals that normalization might lead to incomparable normalized timestamps due to different resolutions (Sections 3.4 and 3.5). Critically, this second aspect is not adhered to by state-of-the-art forensic software, resulting in forensically unsound results (Section 4).

Limitations and Future Work. From a practical standpoint, although our work establishes a theoretical basis for forensically sound timestamp normalization, an implementation is lacking. This should be the subject of future research. Furthermore, timestamp normalization is only one process among many that play a role in timestamp analysis. Therefore, this paper does not address other processes that influence digital investigations in general, or timestamp analysis in particular, which require further formal investigation.

First, the time tracking of digital systems, such as exFAT, can rely on the local time of the device on which it is running. If the local time differs from the actual time, the timestamps must be adjusted to account for this discrepancy [38, 44]. Ideally, future work should specify normalizations that account for discrepancies between local time and actual time by extending proposed methods like using so-called time anchors [43]. Second, obtained timestamps might be ambiguous or incomplete. For example, so-called naive timestamps miss timezone information [26, 37] or unknown date formats may cause dates like 03-02-2026 to be interpreted as either February 3, 2026 or March 2, 2026. To correctly analyze such ambiguous or incomplete timestamps, they must be interpreted in the context of other artifacts such as location data or system data. Future work should address this aspect in order to lay the groundwork for such implementations through a formal model of timestamp contexts. Third, anti-forensics measures like the manipulation of timestamps is a non-negligible possibility that casts doubt on the reliability of digital investigations involving timestamps [32]. Although several methods for detecting tampered timestamps have already been proposed [34], there is no formal framework for verifying obtained timestamps, which ideally aligns the challenge of completing incomplete timestamps. Therefore, future work should aim to establish such a framework for systematically evaluating the trustworthiness of timestamps.

References

- 1 James F. Allen. Maintaining knowledge about temporal intervals. *Commun. ACM*, 26(11):832–843, 1983. doi:10.1145/182.358434.
- 2 Eleftherios Anagnostopoulos, Sotiris Batsakis, and Euripides G.M. Petrakis. CHRONOS: A reasoning engine for qualitative temporal information in OWL. *Procedia Computer Science*, 22:70–77, 2013. doi:10.1016/j.procs.2013.09.082.
- 3 Apple. Apple file system reference. Accessed: March 4, 2026. URL: <https://developer.apple.com/support/downloads/Apple-File-System-Reference.pdf>.
- 4 André Årnes, editor. *Digital Forensics*. Wiley, 1 edition, 2017. doi:10.1002/9781119262442.
- 5 Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. The MIT Press, 2008.
- 6 Claudio Bettini, Sushil Jajodia, and X. Sean Wang. *Time Granularities in Databases, Data Mining, and Temporal Reasoning*. Springer, Berlin, Heidelberg, 2000. doi:10.1007/978-3-662-04228-1.
- 7 Claudio Bettini, X. Sean Wang, and Sushil Jajodia. A general framework for time granularity and its application to temporal reasoning. *Annals of Mathematics and Artificial Intelligence*, 22(1-2):29–58, 1998. doi:10.1023/A:1018938007511.
- 8 Ceri Binding and Douglas Tudhope. Automatic normalization of temporal expressions. *Journal of Computer applications in archaeology*, 6:24–39, 2023. doi:10.5334/jcaa.105.
- 9 Frank Breiting, Hudan Studiawan, and Chris Hargreaves. SoK: Timeline based event reconstruction for digital forensics: Terminology, methodology, and current challenges. *Forensic Science International: Digital Investigation*, 53:301932, 2025. doi:10.1016/j.fsidi.2025.301932.
- 10 Brian Carrier. *File system forensic analysis*. Addison-Wesley Professional, 2005.
- 11 Eoghan Casey. *Digital evidence and computer crime: forensic science, computers and the Internet*. Academic Press, 2011.
- 12 Yoan Chabot, Aurélie Bertaux, Christophe Nicolle, and M-Tahar Kechadi. A complete formalized knowledge representation model for advanced digital forensics timeline analysis. *Digital Investigation*, 11:S95–S105, 2014. doi:10.1016/j.diin.2014.05.009.
- 13 Angel X. Chang and Christopher D. Manning. SUTime: A library for recognizing and normalizing time expressions. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC’12)*, pages 3735–3740. European Language Resources Association (ELRA), 2012. URL: <https://nlp.stanford.edu/pubs/lrec2012-sutime.pdf>.
- 14 E. Ciapessoni, E. Corsetti, A. Montanari, and P. San Pietro. Embedding time granularity in a logical specification language for synchronous real-time systems. *Science of Computer Programming*, 20(1-2):141–171, 1993. doi:10.1016/0167-6423(93)90025-K.
- 15 James Clifford and Ahobala Rao. A Simple General Structure for Temporal Domains. *Temporal Aspects of Information Systems*, pages 17–28, 1988.
- 16 Carlo Combi, Massimo Franceschet, and Adriano Peron. Representing and Reasoning about Temporal Granularities. *Journal of Logic and Computation*, 14(1):51–77, 2004. doi:10.1093/logcom/14.1.51.
- 17 Ana De Almeida Borges, Juan José Conejero Rodríguez, David Fernández-Duque, Mireia González Bedmar, and Joost J. Joosten. To drive or not to drive: A logical and computational analysis of European transport regulations. *Information and Computation*, 280:104636, 2021. doi:10.1016/j.ic.2020.104636.
- 18 Discord. API references. Accessed: March 4, 2026. URL: <https://docs.discord.com/developers/reference>.
- 19 Lisa Marie Dreier, Céline Vanini, Christopher J. Hargreaves, Frank Breiting, and Felix Freiling. Beyond timestamps: Integrating implicit timing information into digital forensic timelines. *Forensic Science International: Digital Investigation*, 49:301755, 2024. doi:10.1016/j.fsidi.2024.301755.
- 20 Jérôme Euzenat and Angelo Montanari. Time Granularity. In *Foundations of Artificial Intelligence*, volume 1, pages 59–118. Elsevier, 2005. doi:10.1016/S1574-6526(05)80005-7.

- 21 Valentin Goranko and Antje Rumberg. Temporal Logic. In Edward N. Zalta and Uri Nodelman, editors, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Summer 2025 edition, 2025.
- 22 The Open Group. Posix.1-2024. Accessed: March 4, 2026. URL: <https://pubs.opengroup.org/onlinepubs/9799919799/>.
- 23 Kristinn Guðjónsson. Mastering the Super Timeline With log2timeline, 2010. URL: <https://www.sans.org/white-papers/33438>.
- 24 International Organization for Standardization (ISO). ISO 8601: Date and time — Representations for information interchange — Part 1: Basic rules, 2019. Accessed: March 4, 2026. URL: <https://www.iso.org/standard/70907.html>.
- 25 International Telecommunication Union (ITU). Recommendation ITU-R TF.460-6: Standard-frequency and time-signal emissions, 2002. Accessed: March 4, 2026. URL: https://www.itu.int/dms_pubrec/itu-r/rec/tf/R-REC-TF.460-6-200202-I!!PDF-E.pdf.
- 26 M. Kaart and S. Laraghy. Android forensics: Interpretation of timestamps. *Digital Investigation*, 11(3):234–248, 2014. doi:10.1016/j.diin.2014.05.001.
- 27 Mohcine Madkour, Driss Benhaddou, and Cui Tao. Temporal data representation, normalization, extraction, and reasoning: A review from clinical domain. *Computer Methods and Programs in Biomedicine*, 128:52–68, 2016. doi:10.1016/J.CMPB.2016.02.007.
- 28 Joachim Metz. Pearls and pitfalls of timeline analysis, 2021. Accessed: March 4, 2026. URL: <https://osdfir.blogspot.com/2021/10/pearls-and-pitfalls-of-timeline-analysis.html>.
- 29 Microsoft. exFAT file system specification. Accessed: March 4, 2026. URL: <https://learn.microsoft.com/en-us/windows/win32/fileio/exfat-specification>.
- 30 Microsoft. File times. Accessed: March 4, 2026. URL: <https://learn.microsoft.com/en-us/windows/win32/sysinfo/file-times>.
- 31 Angelo Montanari. *Metric and Layered Temporal Logic for Time Granularity*. PhD thesis, Universiteit van Amsterdam, 1996. URL: <https://eprints.illc.uva.nl/id/eprint/1995/4/DS-1996-02.text.pdf>.
- 32 Christopher Neale. Fool me once: A systematic review of techniques to authenticate digital artefacts. *Forensic Science International: Digital Investigation*, 45:301516, 2023. doi:10.1016/j.fsidi.2023.301516.
- 33 Netherlands Forensic Institute. Hansken Digital Evidence Platform. Accessed: April 15, 2026. URL: <https://www.hansken.nl/>.
- 34 Junghoon Oh, Sangjin Lee, and Hyunuk Hwang. Forensic Detection of Timestamp Manipulation for Digital Forensic Investigation. *IEEE Access*, 12:72544–72565, 2024. doi:10.1109/ACCESS.2024.3395644.
- 35 Log2Timeline Project. Plaso: Python Logfile Analysis and SuperTimeline Analysis. Accessed: April 15, 2026. URL: <https://github.com/log2timeline/plaso>.
- 36 The Sleuth Kit Project. The Sleuth Kit. Accessed: April 15, 2026. URL: <https://sleuthkit.org/sleuthkit/>.
- 37 Sriram Raghavan and Huzur Saran. UniTIME: Timestamp interpretation engine for developing unified timelines. In *2013 8th International Workshop on Systematic Approaches to Digital Forensics Engineering (SADFE)*, pages 1–7. IEEE, 2013. doi:10.1109/SADFE.2013.6911546.
- 38 Bhupendra Singh and Gaurav Gupta. Analyzing Windows Subsystem for Linux Metadata to Detect Timestamp Forgery. In Gilbert Peterson and Sujeet Shenoi, editors, *Advances in Digital Forensics XV*, volume 569, pages 159–182. Springer International Publishing, 2019. doi:10.1007/978-3-030-28752-8_9.
- 39 Jannik Strötgen and Michael Gertz. Heidelberg: High quality rule-based extraction and normalization of temporal expressions. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 321–324. Association for Computational Linguistics, 2010. URL: <https://aclanthology.org/S10-1071/>.

- 40 Celine Taylor Parkins-Ozephius. Syntax error! Een verkenning van betrouwbaarheidsconcepten bij digitaal bewijs in strafzaken. *Expertise & Recht*, 2025(4), 2025. URL: <https://www.uitgeverijparis.nl/nl/reader/2000000122/2000001179>.
- 41 H.M.A. van Beek, E.J. van Eijk, R.B. van Baar, M. Ugen, J.N.C. Bodde, and A.J. Siemelink. Digital forensics as a service: Game on. *Digital Investigation*, 15:20–38, 2015. doi:10.1016/j.diin.2015.07.004.
- 42 Johan van Benthem. *The Logic of Time*. Kluwer Academic Publishers, 2 edition, 1991.
- 43 Céline Vanini, Christopher J. Hargreaves, Harm Van Beek, and Frank Breiteringer. Was the clock correct? Exploring timestamp interpretation through time anchors for digital forensic event reconstruction. *Forensic Science International: Digital Investigation*, 49:301759, 2024. doi:10.1016/j.fsidi.2024.301759.
- 44 Svein Willassen. Hypothesis-Based Investigation of Digital Timestamps. In Indrajit Ray and Sujeet Sheno, editors, *Advances in Digital Forensics IV*, volume 285, pages 75–86. Springer US, 2008. doi:10.1007/978-0-387-84927-0_7.

A Proofs of Propositions

► **Proposition 5.** *Let D be any time domain, $\preceq$ be its time domain relation, and $\preceq_D^{\mathcal{I}} \subseteq \mathcal{I}_D^{\prec} \times \mathcal{I}_D^{\prec}$ be its interval relation.*

- (1) $\preceq_D^{\mathcal{I}}$ is transitive and antisymmetric.
- (2) If $|D| = 1$, then $\preceq_D^{\mathcal{I}}$ is reflexive and total.
- (3) If $|D| > 1$, then $\preceq_D^{\mathcal{I}}$ is neither reflexive nor total.

Proof. For (1), we need to show that $\preceq_D^{\mathcal{I}}$ is transitive and antisymmetric. For the former, let $[t_1, t_2], [t_3, t_4], [t_5, t_6] \in \mathcal{I}_D^{\prec}$ be arbitrary. Suppose that $[t_1, t_2] \preceq_D^{\mathcal{I}} [t_3, t_4]$ and $[t_3, t_4] \preceq_D^{\mathcal{I}} [t_5, t_6]$. This implies by Definition 4 that $t_2 \preceq t_3$ and $t_4 \preceq t_5$. According to Definition 2, from $[t_3, t_4] \in \mathcal{I}_D^{\prec}$ it follows that $t_3 \preceq t_4$. Consequently, $t_2 \preceq t_3 \preceq t_4 \preceq t_5$. This means according to Definition 4, $[t_1, t_2] \preceq_D^{\mathcal{I}} [t_5, t_6]$. Further, to show that $\preceq_D^{\mathcal{I}}$ is anti-symmetric, let $[t_1, t_2], [t_3, t_4] \in \mathcal{I}_D^{\prec}$ be arbitrary. Suppose that $[t_1, t_2] \preceq_D^{\mathcal{I}} [t_3, t_4]$ and $[t_3, t_4] \preceq_D^{\mathcal{I}} [t_1, t_2]$. From Definitions 2 and 4, the former implies that $t_1 \preceq t_2 \preceq t_3 \preceq t_4$, while the latter implies that $t_3 \preceq t_4 \preceq t_1 \preceq t_2$. Since $\preceq$ is a time domain relation and hence transitive and antisymmetric (see Section 3.1), it follows that $t_1 = t_2 = t_3 = t_4$. This implies that $[t_1, t_2] = [t_3, t_4]$.

For (2), let $|D| = 1$. This means that $\mathcal{I}_D^{\prec} = \{[t, t]\}$ with $t \in D$ and $\preceq_D^{\mathcal{I}} = \{([t, t], [t, t])\}$. It clearly holds that $\preceq_D^{\mathcal{I}}$ is reflexive and total.

For (3), let $|D| > 1$. We need to prove that $\preceq_D^{\mathcal{I}}$ on $\mathcal{I}_D^{\prec}$ is neither reflexive nor total. For the former, it suffices to show that $[t_1, t_2] \in \mathcal{I}_D^{\prec}$ with $[t_1, t_2] \not\preceq_D^{\mathcal{I}} [t_1, t_2]$, while for the latter it suffices to show that $[t_1, t_2], [t'_1, t'_2] \in \mathcal{I}_D^{\prec}$ with $[t_1, t_2] \not\preceq_D^{\mathcal{I}} [t'_1, t'_2]$ and $[t'_1, t'_2] \not\preceq_D^{\mathcal{I}} [t_1, t_2]$. Since $|D| > 1$ and $\preceq$ is a total order, there exist $t_3, t_4 \in D$ with $t_3 \prec t_4$. By Definition 2, $[t_3, t_4] \in \mathcal{I}_D^{\prec}$. Since $t_3 \prec t_4$ and hence $t_4 \not\preceq t_3$, Definition 4 implies that $[t_3, t_4] \not\preceq_D^{\mathcal{I}} [t_3, t_4]$. This means that $\preceq_D^{\mathcal{I}}$ is not reflexive. However, then there exist $[t_1, t_2], [t'_1, t'_2] \in \mathcal{I}_D^{\prec}$ (by picking $[t_1, t_2] = [t'_1, t'_2] = [t_3, t_4]$) such that $[t_1, t_2] \not\preceq_D^{\mathcal{I}} [t'_1, t'_2]$ and $[t'_1, t'_2] \not\preceq_D^{\mathcal{I}} [t_1, t_2]$. Therefore, $\preceq_D^{\mathcal{I}}$ is not total. ◀

► **Proposition 8.** *Let D and T be time domains, $\preceq_D$ and $\preceq_T$ be their time domain order, and $n : D \rightarrow \mathcal{I}_T^{\prec}$ be any normalization of D relative to T . If $D \cap T \neq \emptyset$, then for any $t_1, t_2 \in D \cap T$, $t_1 \preceq_D t_2 \iff n(t_1) \preceq_T n(t_2)$.*

Proof. Suppose that $t_1, t_2 \in D \cap T$. By constraint (iv) of Definition 6, $n(t_1) = [t_1, t_1]$ and $n(t_2) = [t_2, t_2]$. Further, note that according to the specification of time domain orders (Section 3.1), $t_1, t_2 \in D \cap T$ implies that $t_1 \preceq_D t_2$ if and only if $t_1 \preceq_T t_2$.

For the left-to-right direction, suppose that $t_1 \preceq_D t_2$. Then by the specification of time domain orders, $t_1 \preceq_T t_2$. Then by Definition 4 it follows that $[t_1, t_1] \preceq_T^{\mathcal{I}} [t_2, t_2]$. Consequently, $n(t_1) \preceq_T^{\mathcal{I}} n(t_2)$. For the right-to-left direction, suppose that $n(t_1) \preceq_T^{\mathcal{I}} n(t_2)$ and hence $[t_1, t_1] \preceq_T^{\mathcal{I}} [t_2, t_2]$. Definition 4 implies that $t_1 \preceq_T t_2$. By the specification of time domain orders it follows that $t_1 \preceq_D t_2$. $\blacktriangleleft$

► **Proposition 11.** *Let D and T be time domains, while $n : D \rightarrow \mathcal{I}_T^{\mathcal{I}}$ be a non-empty normalization of D relative to T with $\text{Dom}(n) = \{t \in D \mid n(t) \text{ defined}\}$. For any $t \in \text{Dom}(n)$,*

- (1) $D \triangleleft_n^t T$ if and only if $T \not\triangleleft_n^t D$;
- (2) if $T \triangleleft_n^t D$ then $T \preceq_n^t D$;
- (3) if $T \triangleleft_n^t D$ then $D \not\triangleleft_n^t T$;
- (4) if $D \triangleleft_n^{\top} T$ then $T \not\triangleleft_n^{\top} D$;
- (5) if $T \triangleleft_n^{\top} D$ then $T \preceq_n^{\top} D$;
- (6) if $T \triangleleft_n^{\top} D$ then $D \not\triangleleft_n^{\top} T$.

Proof. For proposition (1), note that from Definition 6 it follows that for $t \in \text{Dom}(n)$, $|n(t)| \geq 1$. Consequently, $|n(t)| = 1$ if and only if $|n(t)| \not\geq 1$. By Definition 10 this means that $D \triangleleft_n^t T$ if and only if $T \not\triangleleft_n^t D$.

For proposition (2), suppose that $T \triangleleft_n^t D$ for $t_1 \in \text{Dom}(n)$. According to Definition 10, $T \triangleleft_n^t D$ means that there exists $t_2 \in \text{Dom}(n)$ such that $t_1 \neq t_2$ and $n(t_1) = n(t_2)$. Note that it needs to be shown that $T \not\triangleleft_n^t D$ and hence that $|n(t_1)| = 1$. For the sake of contradiction, assume that $|n(t_1)| > 1$. Therefore, $n(t_1) = [t_3, t_4]$ with $t_3, t_4 \in T$ and $t_3 \neq t_4$. Since $\preceq_T$ is a total order and $t_3 \neq t_4$, $t_3 \prec_T t_4$ or $t_4 \prec_T t_3$. Thus, by Definition 4, $[t_3, t_4] \preceq_T^{\mathcal{I}} [t_3, t_4]$. Further, $n(t_2) = [t_3, t_4]$ because $n(t_1) = n(t_2)$. By constraint (ii) of Definition 6, $n(t_1) \preceq_T^{\mathcal{I}} n(t_2)$ or $n(t_2) \preceq_T^{\mathcal{I}} n(t_1)$ because $t_1 \neq t_2$. This implies that in any of the two case, $[t_3, t_4] \preceq_T^{\mathcal{I}} [t_3, t_4]$. This is a contradiction. Therefore, $|n(t_1)| = 1$ as wanted.

To prove proposition (3), suppose that $T \triangleleft_n^t D$. By proposition (2), $T \preceq_n^t D$. Then by proposition (1), $D \not\triangleleft_n^t T$, as wanted.

To prove (4) - (6), note that n is non-empty. This implies that $|\text{Dom}(n)| \geq 1$. As this holds, (4) follows from (1), (5) follows from (2), and (6) follows from (3). $\blacktriangleleft$

► **Proposition 14.** *Given the time domains D_1, D_2 , and T such that $D_1 = D_2 = T$, then for any normalizations n_1 and n_2 of D_1 and D_2 relative to T respectively, $n_1(t_1) \preceq_T^{\mathcal{I}} n_2(t_2)$ or $n_2(t_2) \preceq_T^{\mathcal{I}} n_1(t_1)$ for any $t_1 \in D_1$ and $t_2 \in D_2$.*

Proof. Let n_1 and n_2 be arbitrary normalization of D_1 and D_2 relative to T , respectively. Let $t_1, t_2 \in D_1 = D_2 = D$ be arbitrary. It suffices to show that $n_1(t_1) \preceq_T^{\mathcal{I}} n_2(t_2)$ or $n_2(t_2) \preceq_T^{\mathcal{I}} n_1(t_1)$. Since $\preceq_T$ on T is a total order, $t_1 \preceq_T t_2$ or $t_2 \preceq_T t_1$. In any of the two cases, Definition 4 implies that $[t_1, t_1] \preceq_T^{\mathcal{I}} [t_2, t_2]$ or $[t_2, t_2] \preceq_T^{\mathcal{I}} [t_1, t_1]$. Since $D_1 = D_2 = T$, constraint (4) of Definition 6 implies that $n_1(t_1) = [t_1, t_1]$ and $n_2(t_2) = [t_2, t_2]$. This means that $n_1(t_1) \preceq_T^{\mathcal{I}} n_2(t_2)$ or $n_2(t_2) \preceq_T^{\mathcal{I}} n_1(t_1)$. $\blacktriangleleft$

► **Proposition 15.** *Let D_1, D_2 , and T be time domains, and n_1 and n_2 be normalization of D_1 and D_2 relative to T , respectively.*

- (1) *If $T \triangleleft_{n_1}^{\top} D_1$ and $T \triangleleft_{n_2}^{\top} D_2$, then $n_1(t_1) \preceq_T^{\mathcal{I}} n_2(t_2)$ or $n_2(t_2) \preceq_T^{\mathcal{I}} n_1(t_1)$ for any $t_1 \in D_1$ and $t_2 \in D_2$ with $n_1(t_1)$ and $n_2(t_2)$ defined.*

- (2) If $T \leq_{n_1}^\top D_1$ and $D_2 \triangleleft_{n_2}^\top T$ where $|n_2(t)| = 2$ for any $t \in \{t \in D_2 \mid n_2(t) \text{ defined}\}$, then $n_1(t_1) \preceq_T^\mathcal{I} n_2(t_2)$ or $n_2(t_2) \preceq_T^\mathcal{I} n_1(t_1)$ for any $t_1 \in D_1$ and $t_2 \in D_2$ with $n_1(t_1)$ and $n_2(t_2)$ defined.

Proof. To prove statement (1), let $t_1 \in \{t \in D_1 \mid n_1(t) \text{ defined}\}$ and $t_2 \in \{t \in D_2 \mid n_2(t) \text{ defined}\}$ be arbitrary. The fact that $T \leq_{n_1}^\top D_1$ and $T \leq_{n_2}^\top D_2$ imply by Definition 10 that $|n_1(t_1)| = 1$ and $|n_2(t_2)| = 1$. This means that $n_1(t_1) = [t_3, t_3]$ and $n_2(t_2) = [t_4, t_4]$ for some $t_3, t_4 \in T$. Since $\preceq_T$ on T is a total order, $t_3 \preceq t_4$ or $t_4 \preceq t_3$. By Definition 4, $[t_3, t_3] \preceq_T^\mathcal{I} [t_4, t_4]$ or $[t_4, t_4] \preceq_T^\mathcal{I} [t_3, t_3]$. It follows that $n_1(t_1) \preceq_T^\mathcal{I} n_2(t_2)$ or $n_2(t_2) \preceq_T^\mathcal{I} n_1(t_1)$.

To prove statement (2), let $t_1 \in \{t \in D_1 \mid n_1(t) \text{ defined}\}$ and $t_2 \in \{t \in D_2 \mid n_2(t) \text{ defined}\}$ be arbitrary. Since $T \leq_{n_1}^\top D_1$, $|n_1(t_1)| = 1$ by Definition 10. This means that $n_1(t_1) = [t_3, t_3]$ for some $t_3 \in T$. In contrast, $|n_2(t)| = 2$ for any $t \in \{t \in D_2 \mid n_2(t) \text{ defined}\}$. Therefore, $n_2(t_2) = [t_4, t_5] = \{t_4, t_5\}$ for some $t_4, t_5 \in D$ with $t_4 \prec_T t_5$ and for any $t \in T$ with $t \neq t_4$ and $t \neq t_5$, $t \prec_T t_4$ or $t_5 \prec_T t$. Since $\preceq_T$ on T is a total order, one of the following cases holds: $t_3 \preceq_T t_4 \prec_T t_5$ or $t_4 \prec_T t_5 \preceq_T t_3$. By Definition 4, it follows that in the first case $n_1(t_1) \preceq_T^\mathcal{I} n_2(t_2)$, while in the second case $n_2(t_2) \preceq_T^\mathcal{I} n_1(t_1)$. ◀

► **Proposition 16.** Let D_1, D_2 and T be time domains, and n_1 and n_2 be gap-free normalizations of D_1 and D_2 relative to T , respectively.

- (1) If $T \leq_{n_1}^\top D_1$, $D_2 \triangleleft_{n_2}^\top T$ where $|n_2(t)| \geq 3$ for any $t \in \{t \in D_2 \mid n_2(t) \text{ defined}\}$, and $|\{t \in T \mid t \in n_1(t'_1) \text{ for some } t'_1 \in D_1\} \cap \{t \in T \mid t \in n_2(t'_2) \text{ for some } t'_2 \in D_2\}| \geq 3$, then there exist $t_1 \in D_1$ and $t_2 \in D_2$ such that $n_1(t_1) \not\preceq_T^\mathcal{I} n_2(t_2)$ and $n_2(t_2) \not\preceq_T^\mathcal{I} n_1(t_1)$.
- (2) If $D_1 \triangleleft_{n_1}^\top T$ where $|n_1(t)| \geq 3$ for any $t \in \{t \in D_1 \mid n_1(t) \text{ defined}\}$, $D_2 \triangleleft_{n_2}^\top T$ where $|n_2(t)| \geq 3$ for any $t \in \{t \in D_2 \mid n_2(t) \text{ defined}\}$, and $|\{t \in T \mid t \in n_1(t'_1) \text{ for some } t'_1 \in D_1\} \cap \{t \in T \mid t \in n_2(t'_2) \text{ for some } t'_2 \in D_2\}| \geq 2$, then there exist $t_1 \in D_1$ and $t_2 \in D_2$ such that $n_1(t_1) \not\preceq_T^\mathcal{I} n_2(t_2)$ and $n_2(t_2) \not\preceq_T^\mathcal{I} n_1(t_1)$.

Proof. For statement (1) note that since $|\{t \in T \mid t \in n_1(t'_1) \text{ for some } t'_1 \in D_1\} \cap \{t \in T \mid t \in n_2(t'_2) \text{ for some } t'_2 \in D_2\}| \geq 3$ and n_1 and n_2 are gap-free, there exist $t_4, t_5, t_6 \in T$ such that (i) $t_4 \prec_T t_5 \prec_T t_6$, (ii) for $t \in \{t_4, t_5, t_6\}$, there exist $t'_1 \in D_1$ and $t'_2 \in D_2$ such that $t \in n_1(t'_1)$ and $t \in n_2(t'_2)$, and (iii) for any $t \in T$ with $t \neq t_4$, $t \neq t_5$, and $t \neq t_6$, $t \prec_T t_4$ or $t_6 \prec_T t$. Further, since $D \leq_{n_1}^\top D_1$, it follows by Definition 10 that $|n_1(t)| = 1$ for any $t \in \{t \in D_1 \mid n_1(t) \text{ defined}\}$. Consequently, there exist $t'_4, t'_5, t'_6 \in D_1$ such that $n_1(t'_4) = [t_4, t_4]$, $n_1(t'_5) = [t_5, t_5]$, and $n_1(t'_6) = [t_6, t_6]$. Further, due to fact that $|n_2(t)| \geq 3$ for any $t \in \{t \in D_2 \mid n_2(t) \text{ defined}\}$, (ii) and (iii), one of following cases holds:

- There exists $t_7 \in D_2$ such that $n_2(t_7) = [t_8, t_9]$ with $t_4, t_5, t_6 \in [t_8, t_9]$. This means that $t_8 \preceq_T t_4$ and $t_6 \preceq_T t_9$. By (i) it follows that $t_8 \prec_T t_5 \prec_T t_9$. Proposition 13 implies that then $n_1(t'_5) \not\preceq_T^\mathcal{I} n_2(t_7)$ and $n_2(t_7) \not\preceq_T^\mathcal{I} n_1(t'_5)$.
- There exist $t_7, t'_7 \in D_2$ such that $n_2(t_7) = [t_8, t_4]$ and $n_2(t'_7) = [t_5, t_9]$ with $t_6 \in [t_5, t_9]$. Since $|n_2(t)| \geq 3$ for any $t \in \{t \in D_2 \mid n_2(t) \text{ defined}\}$, $|n_2(t'_7)| = |[t_5, t_9]| \geq 3$. By (iii), $t_6 \prec_T t_9$. It follows that $t_5 \prec_T t_6 \prec_T t_9$. Proposition 13 implies that then $n_1(t'_6) \not\preceq_T^\mathcal{I} n_2(t'_7)$ and $n_2(t'_7) \not\preceq_T^\mathcal{I} n_1(t'_6)$.
- There exist $t_7, t'_7 \in D_2$ such that $n_2(t_7) = [t_8, t_5]$ and $n_2(t'_7) = [t_6, t_9]$ with $t_4 \in [t_8, t_5]$. By analogous reasoning as in the previous case, $n_1(t'_4) \not\preceq_T^\mathcal{I} n_2(t_7)$ and $n_2(t_7) \not\preceq_T^\mathcal{I} n_1(t'_4)$.

In any of the cases it holds that there exist $t_1 \in D_1$ and $t_2 \in D_2$ with $n_1(t_1) \not\preceq_T^\mathcal{I} n_2(t_2)$ and $n_2(t_2) \not\preceq_T^\mathcal{I} n_1(t_1)$ as wanted.

To prove (2), let us first prove the following statement.

► **Statement 17.** *Let D be a time domain with $\preceq$ being the time domain relation of D . For any $[t_1, t_2], [t_3, t_4] \in \mathcal{I}_D^\preceq$ with $|[t_1, t_2]| \geq 2$, $|[t_3, t_4]| \geq 2$ and $[t_1, t_2] \cap [t_3, t_4] \geq 2$, $[t_1, t_2] \not\preceq_D^{\mathcal{I}} [t_3, t_4]$ and $[t_3, t_4] \not\preceq_D^{\mathcal{I}} [t_1, t_2]$.*

Proof. To prove this, let $[t_1, t_2], [t_3, t_4] \in \mathcal{I}_D^\preceq$ satisfy the aforementioned conditions. Since $|[t_1, t_2] \cap [t_3, t_4]| \geq 2$, there exist $t_5, t_6 \in D$ such that $t_5 \prec t_6$ with $t_5, t_6 \in [t_1, t_2]$ and $t_5, t_6 \in [t_3, t_4]$. Further, since $|[t_1, t_2]| \geq 2$ and $|[t_3, t_4]| \geq 2$, $t_1 \preceq t_5 \prec t_6 \preceq t_2$ and $t_3 \preceq t_5 \prec t_6 \preceq t_4$. Suppose towards contradiction that $[t_1, t_2] \preceq_D^{\mathcal{I}} [t_3, t_4]$. This implies that $t_5 \prec t_6 \preceq t_2 \preceq t_3 \preceq t_5 \prec t_6$. So that $t_6 \preceq t_5$ and $t_6 \not\preceq t_5$ (because $t_5 \prec t_6$) which is a contradiction. Therefore, $[t_1, t_2] \not\preceq_D^{\mathcal{I}} [t_3, t_4]$. By analogous reasoning, one obtains that $[t_3, t_4] \not\preceq_D^{\mathcal{I}} [t_1, t_2]$. ◀

Given this result, we can then prove (2). Note that since $|\{t \in T \mid t \in n_1(t'_1) \text{ for some } t'_1 \in D_1\} \cap \{t \in T \mid t \in n_2(t'_2) \text{ for some } t'_2 \in D_2\}| \geq 2$ and n_1 and n_2 are gap-free, there exist $t_4, t_5 \in T$, such that (i) $t_4 \prec_T t_5$, (ii) for $t \in \{t_4, t_5\}$, there exist $t'_1 \in D_1$ and $t'_2 \in D_2$ such that $t \in n_1(t'_1)$ and $t \in n_2(t'_2)$, and (iii) for any $t \in T$ with $t \neq t_4$ and $t \neq t_5$, $t \prec_T t_4$ or $t_5 \prec_T t$. This implies that one of the following cases holds:

- There exist $t_7 \in D_1$ and $t_8 \in D_2$ such that $t_4, t_5 \in n_1(t_7)$ and $t_4, t_5 \in n_2(t_8)$. By Statement 17, it then follows that $n_1(t_7) \not\preceq_T^{\mathcal{I}} n_2(t_8)$ and $n_2(t_8) \not\preceq_T^{\mathcal{I}} n_1(t_7)$.
- There exist $t_7, t'_7 \in D_1$ and $t_8 \in D_2$ such that $n_1(t_7) = [t_9, t_4]$, $n_1(t'_7) = [t_5, t_{10}]$ and $t_4, t_5 \in n_2(t_8)$. Since n_1 and n_2 are gap-free, $|n_1(t)| \geq 3$ for any $t \in \{t \in D_1 \mid n_1(t) \text{ defined}\}$, and $|n_2(t)| \geq 3$ for any $t \in \{t \in D_2 \mid n_2(t) \text{ defined}\}$, $|n_1(t_7) \cap n_2(t_8)| \geq 2$ or $|n_1(t'_7) \cap n_2(t_8)| \geq 2$. It follows by Statement 17 that $n_1(t_7) \not\preceq_T^{\mathcal{I}} n_2(t_8)$ and $n_2(t_8) \not\preceq_T^{\mathcal{I}} n_1(t_7)$, or $n_1(t'_7) \not\preceq_T^{\mathcal{I}} n_2(t_8)$ and $n_2(t_8) \not\preceq_T^{\mathcal{I}} n_1(t'_7)$.
- There exist $t_7 \in D_1$ and $t_8, t'_8 \in D_2$ such that $t_4, t_5 \in n_1(t_7)$, $n_2(t_8) = [t_{11}, t_4]$, and $n_2(t'_8) = [t_5, t_{12}]$. By analogous reasoning as in the previous case, $n_1(t_7) \not\preceq_T^{\mathcal{I}} n_2(t_8)$ and $n_2(t_8) \not\preceq_T^{\mathcal{I}} n_1(t_7)$, or $n_1(t_7) \not\preceq_T^{\mathcal{I}} n_2(t'_8)$ and $n_2(t'_8) \not\preceq_T^{\mathcal{I}} n_1(t_7)$.
- There exist $t_7, t'_7 \in D_1$ and $t_8, t'_8 \in D_2$ such that $n_1(t_7) = [t_9, t_4]$, $n_1(t'_7) = [t_5, t_{10}]$, $n_2(t_8) = [t_{11}, t_4]$, and $n_2(t'_8) = [t_5, t_{12}]$. By analogous reasoning as in the previous cases, $n_1(t_7) \not\preceq_T^{\mathcal{I}} n_2(t_8)$ and $n_2(t_8) \not\preceq_T^{\mathcal{I}} n_1(t_7)$, and $n_1(t'_7) \not\preceq_T^{\mathcal{I}} n_2(t'_8)$ and $n_2(t'_8) \not\preceq_T^{\mathcal{I}} n_1(t'_7)$.

Therefore, in any of the cases it holds that there exist $t_1 \in D_1$ and $t_2 \in D_2$ with $n_1(t_1) \not\preceq_D^{\mathcal{I}} n_2(t_2)$ and $n_2(t_2) \not\preceq_T^{\mathcal{I}} n_1(t_1)$ as wanted. ◀