A PROOF

Table 3 lists the frequently used notations.

Table 3: Notations

Symbol	Description
$\mathcal{E}$	a set of events E
$t[E]$	timestamp of an event E in a tuple (trace) t over $\mathcal{E}$
p_i	event pattern (Definition 1)
$t[p_0^s]$, $t[p_0^e]$	start / end timestamp of p_0 in a tuple (trace) t
$t \vDash p_0$	a tuple (trace) t matches pattern p_0 (Definition 2)
$\phi(E_i, E_j)$:$[a, b]$	interval conditions between timestamps of E_j and E_i, $a \le t[E_j] - t[E_i] \le b$ (Definition 3)
$t \vDash \Phi$	a tuple (trace) t matches a set of interval conditions Φ
$\gamma(E_i, \mathcal{E}_i)$:$min$	binding condition on timestamps of events in $\mathcal{E}_i$, i.e., $t[E_i] = \min_{E_j \in \mathcal{E}_i} t[E_j]$ (Definition 4)
$t \vDash \Gamma$	a tuple (trace) t matches a set of binding conditions Γ
$\aleph_\Gamma$	all the possible bindings for Γ

A.1 Proof of Proposition 1

Referring to Definition 1, a pattern p_0 is declared over a set of events, say $\mathcal{E}_0$. It applies to the (at least two) sub-patterns $p_1, \ldots, p_k$ embedded in SEQ or AND of p_0, denoted by $\mathcal{E}_1, \ldots, \mathcal{E}_k$. Referring to the sequence/concurrent relationships among $p_1, \ldots, p_k$, we have $\mathcal{E}_i \cap \mathcal{E}_j = \emptyset, i, j \in 1, \ldots, k$, i.e., the sub-patterns $p_1, \ldots, p_k$ do not overlap on events. It follows that the number of sub-patterns embedded in p_0 is not greater than $|\mathcal{E}_0|$. As presented in Definition 2, the [ATLEAST a] [WITHIN b] conditions in each pattern can be checked in $O(k)$ time. The time cost of determination of whether $t \vDash p_0$ is thus $O(|\mathcal{E}_0|^2)$, where $|\mathcal{E}_0|$ is the number of events declared in p_0.

A.2 Proof of Theorem 2

The problem is clearly in NP. For any tuple t, whether it matches each pattern can be recursively checked, referring to Proposition 1.

To show the hardness of the pattern consistency problem, we present a reduction from the 3SAT problem, which is one of Karp's 21 NP-complete problems [27]. Consider a CNF formula $C = c_1 \wedge \cdots \wedge c_m$, where each clause $c_i = l_{i1} \vee l_{i2} \vee l_{i3}$ have three literals, $i = 1, \ldots, m$, and each literal l_{ij} is either x or $\neg x$ for some variable $x \in X, n = |X|$. The 3SAT problem is to determine whether there exists an assignment to make all the clauses satisfied.

The transformation first introduces a set of events

$$\mathcal{E} = \{C_0, C_1, \ldots, C_m, X_1, \ldots, X_n, \neg X_1, \ldots, \neg X_n\},$$

where each C_i corresponds to a clause $c_i, i = 1, \ldots, m$, and $X_j, \neg X_j$ represent $x_j, \neg x_j$, respectively, $j = 1, \ldots, n$. A set $\mathcal{P}$ of event patterns over $\mathcal{E}$ are constructed as follows:

(1) For each $x_j, \neg x_j$, we introduce an event pattern

p_j :SEQ(C_0, AND($X_j, \neg X_j$) ATLEAST 1 WITHIN 1) ATLEAST 3 WITHIN 3.

(2) For each clause $c_i = l_{i1} \vee l_{i2} \vee l_{i3}$, we add an event pattern

p_{n+i} : SEQ(C_i, AND(X_{i1}, X_{i2}, X_{i3})) ATLEAST 2 WITHIN 2.

(3) For each C_i, we consider

SEQ(C_0, C_i) ATLEAST 1 WITHIN 1.

Event pattern p_j requires that the timestamp distance between X_j and $\neg X_j$ is exactly 1, while p_{n+i} requires that at least one of X_{i1}, X_{i2}, X_{i3} should have timestamp distance 2 compared to C_i. (Figure 14 illustrates the interval conditions in the complex temporal networks corresponding to the aforesaid patterns, see Definition 5 for details.)

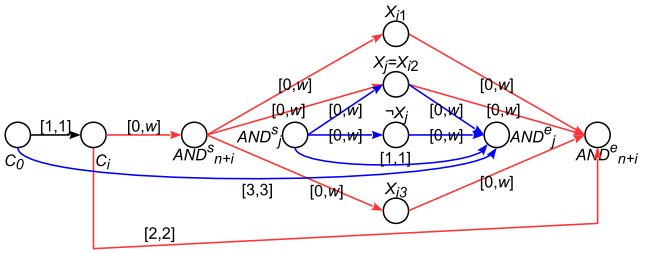

Figure 14: Reduction from 3SAT problem

We show that the CNF formula in transformation is satisfiable if and only if there exists a tuple t of events such that $t[C_0] = s, t[C_i] = s + 1, t[X_{ik}] = s + 3$ for some $X_{ik}, k = 1, 2, 3, i = 1, \ldots, n$, s is any timestamp in the domain, i.e., $t \vDash \mathcal{P}$ the event patterns are consistent.

First, if the CNF formula in transformation is satisfiable, we set the corresponding timestamp assignment $t[C_i] = s + 1$ and $t[C_0] = s$, where s is any timestamp in the domain. For each clause $c_i = l_{i1} \vee l_{i2} \vee l_{i3}$, there must exist some $l_{ik} = \text{true}, k = 1, 2, 3$, while others equal 0. For each $l_{ik} = \text{true}$, we assign $t[X_{ik}] = s + 3$; otherwise, $t[X_{ik}] = s + 2$ for $l_{ik} = \text{false}$. It is easy to see $t \vDash p_{n+i}$ in $\mathcal{P}$ in the transformation. Moreover, for each $X_j = \text{true}$, we have $t[X_j] = s+3$ and $t[\neg X_j] = s + 2$; otherwise, $t[X_j] = s + 2$ and $t[\neg X_j] = s + 3$ for $X_j = \text{false}$. The CNF assignment guarantees that $t \vDash p_j$ in $\mathcal{P}$ in the transformation. That is, we have $t \vDash \mathcal{P}$.

Conversely, suppose that the event patterns are consistent, i.e., there exists a timestamp assignment t having $t \vDash \mathcal{P}$. Let $t[C_0] = s$ the timestamp in the domain. It follows $t[C_i] = s + 1, i = 1, \ldots, m$. Pattern p_j in $\mathcal{P}$ in the transformation ensures that only one of $X_j, \neg X_j$ will have t label $s + 3$, i.e., true in the CNF assignment; the other will have t label $s + 2$ and false assignment in the CNF. Referring to $t \vDash p_{n+i}$ in $\mathcal{P}$ in the transformation, for each $i = 1, \ldots, n$, there must exists some $X_{ik}, k = 1, 2, 3$ such that $t[X_{ik}] = s + 3$. According to the aforesaid assignment, we have the corresponding literal $l_{ik} = \text{true}$. For others with $t[X_{ik}] = s + 2$, it has $l_{ik} = \text{false}$. That is, the CNF formula is satisfiable.

A.3 Proof of Theorem 3

The problem is clearly in NP. Given a tuple t' with modified timestamp, whether it matches each pattern can be recursively checked, referring to Definition 2, and the modification cost can be calculated in $O(n)$ time by Formula 1.

To illustrate the hardness of the timestamp modification problem, we show a reduction from the SET COVER problem, which is one of Karp's 21 NP-complete problems [27].

Given a set of m elements $\mathcal{U} = \{u_1, \ldots, u_m\}$ and n sets $\mathcal{S} = \{s_1, \ldots, s_n\}$ such that $s_i \subseteq \mathcal{U}$ and $\cup_i s_i = \mathcal{U}$. A set cover is a $C \subseteq \mathcal{S}$ of sets whose union is still $\mathcal{U}$. The minimum set cover problem is to identify the smallest number of sets whose union still contains all elements in $\mathcal{U}$.

Consider a set of events

$$\mathcal{E} = \{S_1, \ldots, S_i, \ldots, S_n, S'_1, \ldots, S'_n, U_1, \ldots, U_j, \ldots, U_m\}$$

where each S_i, S'_i represents a set s_i, and U_j corresponds to an element u_j. Let $\mathcal{E}_j = \{S_i \in \mathcal{E} \mid u_j \in s_i\}$, i.e., all the events S_i whose corresponding s_i contains u_j.

First, the event pattern set $\mathcal{P}$ is constructed as follows.

(1) For each U_j, we add into $\mathcal{P}$ a pattern

$$p_j : \texttt{SEQ}(U_j, \texttt{AND}(S_{i1}, \ldots, S_{ik}))\ \texttt{ATLEAST 2 WITHIN 2},$$

where $S_{i1}, \ldots, S_{ik}$ correspond to all the sets $s_{i1}, \ldots, s_{ik} \subseteq \mathcal{S}$ containing element u_j.

(2) For each U_j, S'_i, we introduce a pattern

$$\texttt{SEQ}(S'_i, U_j)\ \texttt{ATLEAST 1 WITHIN 1}.$$

(Figure 15 illustrates the interval conditions in the complex temporal networks corresponding to the aforesaid patterns, see Definition 5 for details.)

Next, we build a tuple t of events where $t[S'_i] = 0, t[U_j] = 1, t[S_i] = 2, i = 1, \ldots, n, j = 1, \ldots, m$. The patterns require that there must exist a $S_i \in \mathcal{E}_j$ such that $t[S_i] - t[U_j] = 2$, which is not observed in the given t.

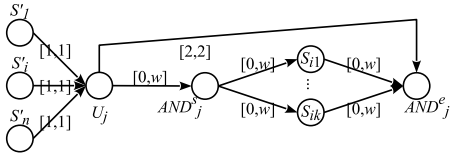

Figure 15: Reduction from SET COVER problem

To prove NP-hardness we show that there is a set cover C of size k if and only if there exists a tuple t' matches the event patterns in the transformation and $\Delta(t, t') = k$.

First, if there exists a set cover C with size k, for each $s_i \in C$, we assign the corresponding $t'[S_i] = 3$. Referring to the modification cost in Formula 1, we have $\delta(t', t) = k$. Moreover, the set cover requires that for each u_j, there is at least one set s_i in C that contains u_j, i.e., having the corresponding $t'[S_i] = 3$. It leads to $t' \vDash \mathcal{P}$.

Suppose that there is a tuple t' with $t' \vDash \mathcal{P}$ and $\Delta(t, t') = k$. Referring to the transformation of each pattern p_j in $\mathcal{P}$, there must have some $S_i \in \mathcal{E}_j$ with modified $t'[S_i] = 3$. By collecting all the corresponding s_i, since all the elements u_j are covered, it forms a set cover C with size k.

A.4 Proof of Theorem 4

To show the hardness of approximation, we illustrate that the reduction in Theorem 3 from the SET COVER problem, which is known NP-hard to approximate to within any constant factor [11], is gap-preserving.

Let C^* denote a minimum set cover, and t^* be a modified tuple with the minimum modification cost. It is sufficient to conclude

- if $|C^*| \le k$, then $\Delta(t, t^*) \le k$,
- if $|C^*| > \alpha k$, then $\Delta(t, t^*) > \alpha k$,

where $\alpha > 1$.

A.5 Proof of Proposition 5

The conclusion can be proved by induction referring to the recursive Definitions 2 and 5.

(1) For $p = E$, we have $t[p^s] = t[p^e] = t[E]$ for any t in Definition 2, and $\Phi_p = \emptyset$, $\Gamma_p = \emptyset$, referring to item 1 in Definition 5.

(2) For $p = \texttt{SEQ}(p_1, p_2, \ldots, p_k)$ [ATLEAST a] [WITHIN b], referring to item 2 in Definition 5, it has $\Gamma_p \setminus (\Gamma_{p_1} \cup \cdots \cup \Gamma_{p_k}) = \emptyset$. First, suppose that $t \vDash p$. Referring to $t[p_i^e] < t[p_{i+1}^s]$ in item 2 in Definition 2, we have $t \vDash \phi(E^e_{p_i}, E^s_{p_{i+1}})$:$[1, w], i = 1, \ldots, k-1$, in item 2 in Definition 5. Moreover, the ATLEAST a WITHIN b condition is equivalent to the interval condition $\phi(E^s_p, E^e_p)$:$[a, b]$. It leads to $t \vDash (\Phi_p, \Gamma_p)$. Conversely, we show $t \vDash (\Phi_p, \Gamma_p)$ implies $t \vDash p$ in a similar way.

(3) For $p = \texttt{AND}(p_1, p_2, \ldots, p_k)$ [ATLEAST a] [WITHIN b], the binding condition $\gamma(E^s_p, \{E^s_{p_1}, \ldots, E^s_{p_k}\})$:$min$ and the interval conditions $\phi(E^s_p, E^s_{p_i})$:$[0, w], i = 1, \ldots, k$ in item 3 in Definition 5 ensure $t[p^s] = \min(\pi(p_1^s), \pi(p_2^s), \ldots, \pi(p_k^s))$ in item 3 in Definition 2. Similar equivalence on max binding condition can also be observed. Again, the ATLEAST a WITHIN b condition is equivalently captured by the interval condition $\phi(E^s_p, E^e_p)$:$[a, b]$.

A.6 Proof of Corollary 6

Similar to the proof of Proposition 5, the conclusion can be proved by induction referring to the recursive Definitions 2 and 6. In particular, since there is no binding condition, the proof w.r.t. binding conditions in step (3) in the proof of Proposition 5 is not necessary.

A.7 Proof of Proposition 7

The binding conditions Γ appear only w.r.t. AND event patterns, together with a list of interval conditions, referring to item 3 in Definition 5.

First, suppose that there exists a $\Phi_k \in \aleph_\Gamma$ such that $\Phi \cup \Phi_k$ is consistent, i.e., having some $t \vDash \Phi \cup \Phi_k$. For each binding condition (say $\gamma(E_i, \mathcal{E}_i)$:$min$) in Γ, we have some $E_j \in \mathcal{E}_i$ with $t[E_i] = t[E_j]$, according to the full binding in Definition 7. Since $t \vDash \Phi$, we have $t[E_i] = t[E_j] \le t[E_k], \forall E_k \in \mathcal{E}_i$, referring to the aforesaid interval conditions in item 3 in Definition 5. That is, it has $t \vDash \Gamma$ as well.

Next, suppose that (Φ, Γ) is consistent, having some $t \vDash (\Phi, \Gamma)$. For each binding condition (say $\gamma(E_i, \mathcal{E}_i)$:$min$) in Γ, we have some $E_j \in \mathcal{E}_i$ with $t[E_i] = t[E_j]$, according to the binding condition Definition 4. That is, it satisfies $t \vDash \phi(E_i, E_j)$:$[0, 0]$ in Φ_γ. Considering the aforesaid interval condition for each binding condition γ in Γ, we obtain a $\Phi_k \in \aleph_\Gamma$ having $t \vDash \Phi_k$.

A.8 Proof of Proposition 8

For the events $\mathcal{E} = \{E_1, \ldots, E_n\}$ in the pattern, we denote $E_{\max} = \arg\max_{E \in \mathcal{E}} t[E]$ and $E_{\min} = \arg\min_{E \in \mathcal{E}} t[E]$. Referring to the single binding in Definition 8, we have

$$\Phi_1 = \{\phi(\texttt{AND}^s, E_{\min})\text{:}[0, 0], \phi(\texttt{AND}^e, E_{\max})\text{:}[0, 0]\}.$$

Let t^* be the optimal timestamp modification w.r.t. full binding, having $E^*_{\max} = \arg\max_{E\in\mathcal{E}} t^*(E)$, and $E^*_{\min} = \arg\min_{E\in\mathcal{E}} t^*(E)$. We show that t^* is also a timestamp modification w.r.t. the single binding, in all the possible cases.

(1) If $t[E_{\min}] \le t^*[E^*_{\min}]$, it has $t^*[E] = t^*[E^*_{\min}]$ for all $E \in \mathcal{E} \setminus \{E_{\max}\}$ with $t[E] \le t^*[E^*_{\min}]$. Any other possible timestamp modification on E greater than $t^*[E^*_{\min}]$ must have modification cost no less than π^*. It indicates $t^*[E_{\min}] = t^*[E^*_{\min}]$.

(2) If $t[E_{\min}] > t^*[E^*_{\min}]$, it has $t^*[E_{\min}] = t^*[E^*_{\min}]$. Any other possible modification with a different event having the minimum timestamp $t^*[E^*_{\min}]$ must have modification cost no less than t^*. It also indicates $t^*[E_{\min}] = t^*[E^*_{\min}]$.

(3) If $t[E_{\max}] \ge t^*[E^*_{\max}]$, it has $t^*[E] = t^*[E^*_{\max}]$ for all $E \in \mathcal{E} \setminus \{E_{\min}\}$ with $t[E] \ge t^*[E^*_{\max}]$. Any other possible timestamp modification on E less than $t^*[E^*_{\max}]$ must have modification cost no less than t^*. It indicates $t^*[E_{\max}] = t^*[E^*_{\max}]$.

(4) If $t[E_{\max}] < t^*[E^*_{\max}]$, it has $t^*[E_{\max}] = t^*[E^*_{\max}]$. Any other possible modification with a different event having the maximum timestamp $t^*[E^*_{\max}]$ must have modification cost no less than t^*. It also indicates $t^*[E_{\max}] = t^*[E^*_{\max}]$.

Since we have $t^*[E_{\max}] = t^*[E^*_{\max}]$ and $t^*[E_{\min}] = t^*[E^*_{\min}]$ in all the cases, it is sufficient to show that t^* is also a timestamp modification w.r.t. the single binding Φ_1, and could be returned by Algorithm 2 with single binding.

B ADDITIONAL RESULTS

B.1 Pattern Set for Consistency Evaluation

For $n = 1$, the event pattern set is defined by an AND pattern

$$p_1 : \texttt{AND}(\texttt{SEQ}(E_{11}, E_{12})\ \texttt{ATLEAST 1}, \texttt{SEQ}(E_{13}, E_{14})\ \texttt{ATLEAST 1})\ \texttt{ATLEAST 1 WITHIN } b,$$

and a SEQ pattern $p_2 : \texttt{SEQ}(E_{11}, E_{14})\ \texttt{ATLEAST 0 WITHIN 0}$. The first pattern p_1 with parallel and sequential events corresponds to the edges A->B->C->D and A->E->F->D, as illustrated in the Figure 16(a) , while the second pattern p_2 is denoted by edge B->F.

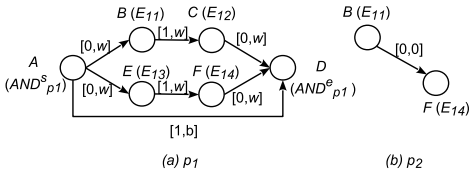

Figure 16: Example pattern set when n = 1

B.2 Distribution of Actual Flight Durations

Figure 17 illustrates the distribution of the actual flight durations.

C DATABASE QUERY EXPLANATIONS

The "why not" explanations of database queries, i.e., to explain why a query result is missing [14], have been widely studied, mainly in two different aspects.

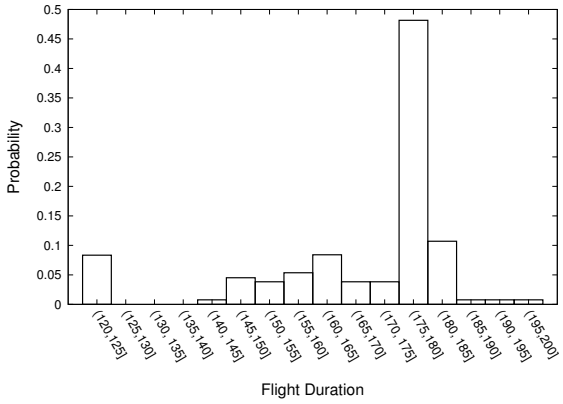

Figure 17: Probability distribution of flight duration from EWR to MCO without a stopover

C.1 Query Explanation

The first category focuses on the issues in queries for the explanation. Chapman and Jagadish [14] pinpoint the manipulation that discarded the desired tuple from the final result set to answer why not problems for the SQL queries on relational data. However, they assume that no conflict occurs among the conditions in SQL queries, i.e., the SQL queries are consistent. In this work, when facing a missing result, we first refer to the pattern consistency explanation to check if there exists any tuple (trace) matching the given patterns. Different from [14] that just points out the manipulation accounting for the non-answers, Gao et al. [20] revise the initial query to generate a refined reverse top-k query whose result contains the user specified missing tuples. Bao et al. [9] combine both ideas by giving the constraints leading to mismatch as well as the suggestion to modify the original queries. They explain the mismatch problem in the context of XML keyword search by finding the constraints causing the empty result, and generate some alternative constraints to form a suggested query. The aforesaid techniques for XML keyword search are not applicable to the pattern queries over event tuples.

C.2 Data Explanation

The second category focuses on the issues in the data for the explanation. Huang et al. [24] look for the attributes that cause the data item of interest to be excluded. Herschel and Hernández [23] provide a set of data modifications so that the missing tuples can present in the query result. Since event pattern queries on event data are different from SQL queries on relational data, the existing techniques cannot be applied directly to our problem. Nevertheless, following the same line, we propose to modify the timestamps of events, so that the modified tuple (trace) could match the given patterns. Lee et al. [28] use sampling to generate summaries of potential data instances for answering that balance the completeness and informativeness, to overcome the scalability challenge that the provenance could be very large. In order to minimize the number of explanations for the missing results, in this work, we return the modified tuple (trace) with the minimum timestamp modification

cost as the explanation. Explanations for the interesting or unexpected trends and anomalies in data are explored in [32]. After pre-computing the effects of potential aggregation results on data, the explanation to the specific query can be computed at an interactive speed by efficiently re-evaluate the original query taking into account these effects. Different from SQL queries, no aggregation results could be pre-computed in the event pattern queries.

C.3 Pattern Consistency Checking

Consistency checking in Section 2.2 for pattern consistency explanation is related to the temporal constraint satisfaction problem (TCSP) [12]. Evolutionary algorithms enhanced by heuristic and adaptive fitness computation are applied to deal with TCSP in [25]. Filtering algorithms are proposed in [15] to reduce the searching domains of the variables in the preprocessing step for solving TCSP. However, `AND` pattern with `ATLEAST` predicate is not studied in TCSP in the aforesaid studies. Indeed, as shown in Table 2 and Section 4.1, event patterns without `ATLEAST` or `AND` are simple temporal networks [16], where the above work could be applied. Pralet and Verfaillie [31] extend the simple temporal constraints to shift monotonic constraints, where the constraints defined between variables are time-dependent instead of constant numbers. However, they still focus on the `SEQ` predicate, while our study considers the more complicated event patterns with `AND` predicate.