A. Appendix: application of our temporal motifs

There are certain potential applications of our defined temporal motifs in temporal networks, where the nodes and edges are fixed, but the edge labels vary regularly with timestamps [1]–[4].

(1) Road traffic analyses. There are many sensors, cameras and floating cars in road networks collecting road traffic information periodically [5]. In contrast to dynamic traffic information, roads are relatively fixed, which can be naturally represented as structurally-static temporal networks with edge labels varying regularly with timestamps [2], [6], [7]. Here, edge labels can represent real-time reports of traffic condition on roads, such as congestion levels (*e.g.,* congested, slow, and fast). In such a temporal network, our temporal motifs can discover different kinds of traffic patterns, including long-time congestion patterns with all congested roads, congestion propagation patterns with bottlenecks and influenced roads and hybrid traffic condition patterns [8], [9]. Understanding these traffic patterns helps to improve the traffic conditions by rearranging traffic policemen or redesigning tidal lanes and traffic lights.

(2) Energy consumption analyses. There are many open energy systems collecting information of transmission networks, energy generation and demand signals [10]–[12]. A structurally-static temporal network can be built to represent the dynamics of those energy signals, where each transmission line corresponds to an edge, and the merging point of transmission lines corresponds to a node. Each edge or node can be labeled by the corresponding series of energy signal levels in the time interval of the temporal network. In such a temporal network, our temporal motifs can discover different kinds of energy consumption patterns, including long-term high energy consumption patterns and hybrid energy consumption patterns [5], [11], [12]. Long-term high energy consumption patterns correspond to regions with continuous high energy demand signal levels, and hybrid energy consumption patterns correspond to regions with both continuous high and low energy demand signal levels. Understanding these energy consumption patterns helps to reveal the implied correlation among energy demand, regions and seasons, and to design proper energy preservation and transmission strategies.

(3) Flood risk analyses. There are many wireless sensor network systems with different types of sensors recording significant environment information, including the water discharge from dam, rainfall, humidity and temperature [13]–[15]. With the combination of the geographical information system [14], [15], a structurally-static temporal network can be built, where the sub-region corresponds to a node and each adjacency relation between sub-regions corresponds to an edge. Each edge can be labeled by the corresponding series of water peak discharge levels, storm levels, etc, which are typically averaged over each sub-region [13]. In such a temporal network, our temporal motifs can discover different kinds of environment patterns with continuous low or high water and weather signal levels. Understanding these patterns helps to identify flood risk areas in order to take necessary actions before the flood occurs [15], and to alleviate the risk by redesigning the water discharge from dam.

B. Appendix: correctness proofs of temporal motif analyses

Proof of Proposition 1: Assume that there exists an edge e in $\mathsf{R}[$ m, i] and $\mathsf{R}[$ m,j] ($m+k-1 \leq i < j \leq T$). By the definition, we have $e \in \mathsf{S}[m,i]-\mathsf{S}[m,i+1]$ and $e \in \mathsf{S}[m,j]-\mathsf{S}[m,j+1]$ or $\mathsf{S}[m,T]$ if $j=T$. However, as $i<j$ and $\mathsf{S}[m,j] \subseteq \mathsf{S}[m,i+1]$, $e \in \mathsf{S}[m,i+1]$. This implies $e \notin \mathsf{R}[m,i]$, which is a contradiction. Hence we have the conclusion. □

Proof of Proposition 2: Assume that there is a connected component that is not a maximal temporal motif $G_s[V_s, E_s, m, i]$. It implies that there is an extra edge $e \notin E_s$ adjacent to a node in G_s and having the same label in interval $[m,i]$, which contradicts with the fact that connected components are maximal [16]. Hence we have the conclusion. □

Proof of Proposition 3: Assume that we are generating the maximal temporal motifs for interval $[m,i]$ with edge set $\mathsf{R}[m,i]$ ($i \in [m+k-1,T]$). By the definition of R edge sets, any edge in $\mathsf{R}[m,i]$ changes its label at timestamp $i+1$. Hence, we have the conclusion. □

C. Appendix: correctness proofs of procedure computeRES

Proof of Proposition 4: We assume that edge e has the same label in each of the intervals $[H_1=1,U_1], \cdots, [H_n,U_n=T]$ such that $H_{i+1}=U_i+1$ for $1 \leq i < n$, and that e has different labels in any two adjacent intervals. The length of $[H_i,U_i]$ is also denoted as l_i for $1 \leq i \leq n$. Note that $n=|V_e|$ is the number of nodes for the IC tree of e.

There are two cases for $[H_i,U_i]$ during the computation of R edge sets.

(1) Case $l_i \geq k$. For the H_i-th row, the EL table or the IC tree of e needs to be used (cases (1) and (2)), which takes l_i-k and $\log n$ time, respectively. For rows from H_i+1 to U_i, $\mathsf{EMaxIntvl}[e]$ is used once (case (4)), which takes constant time 1 for each row.

(2) Case $l_i < k$. For rows from H_i to U_i, the EL table or IC tree of e needs to be used (cases (1) and (2)), which takes constant time 1 and $\log n$ time for each row, respectively.

For the EL table, the worst case for dealing with $[H_i,U_i]$ is $2l_i-k-1$ time, and the following holds for T rows: $\sum_{i=1}^{n}(2l_i-k-1) < \sum_{i=1}^{n}(2l_i) = 2T$. For IC trees, the worst case for dealing with $[H_i,U_i]$ is $l_i \log n$ time, and the following holds for T rows: $\sum_{i=1}^{n}(l_i \log n) = T \log n$. As there are in total T rows, we have the conclusion. □

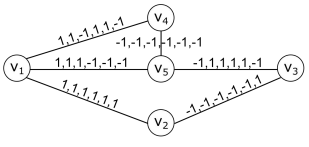
D. Appendix: detailed process of algorithm FTM

The complete algorithm FTM is shown in Fig.1, which takes as input a temporal graph $G(V,E,1,T,L)$ and a frequency

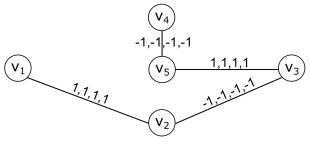
Algorithm FTM
Input: Temporal graph $G(V, E, 1, T, L)$ and frequency threshold k.
Output: Set TF of all the maximal and non-expandable temporal motifs.

1. **let** ds be the EL table or IC trees for G;
2. EMaxIntvl$[e]$:= null for each e;
3. **for** $m := 1$ **to** $T - k + 1$ /*T2B*/
4. (R, EMaxIntvl) := computeRES(ds, k, EMaxIntvl, m);
5. CC$[i, T] := \emptyset$ for $i \in [m + k - 1, T + 1]$;
6. **for** $i := T$ **to** $m + k - 1$ /*R2L/
7. CC$[i, T]$:= generateMaxTM(R, EMaxIntvl, $[m, i]$, CC$[i + 1, T]$); /*compute maximal motifs with the interval $[m, i]$*/
8. TF$[m, i]$:= generateExpTM(CC$[i, T]$, $[m, i]$);
9. **return** TF.

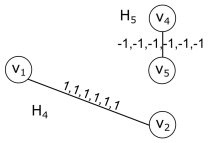
Fig. 1: Algorithm FTM



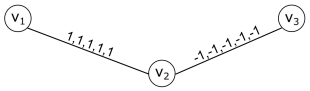
(a) Temporal graph G with interval $[1, 6]$



(b) Temporal subgraph H_1 of G with interval $[2, 5]$



(c) Temporal subgraphs H_4 and H_5 of G with interval $[1, 6]$



(d) Temporal subgraph H_6 of G with interval $[1, 5]$

Fig. 2: Running example

threshold k, and returns the set TF of all the maximal and non-expandable temporal motifs. It first creates the EL table or IC trees (referred to as ds) (line 1), and initializes the array EMaxIntvl to be empty (line 2). Then it adopts the T2B scheme to deal with all rows m from 1 to $T-k+1$ (line 3). For each row m, it invokes computeRES to obtain the edge set R$[m, i]$ for $m+k-1 \leq i \leq T$ and the updated array EMaxIntvl (line 4), and initializes connected components CC$[i, T]$ for $i \in [m+k-1, T+1]$ (line 5). Note that we use CC$[T+1, T]$ to represent an empty set for convenience. After that, it adopts the R2L scheme with columns i from T to $m+k-1$ (line 6). For each i, it invokes generateMaxTM to obtain the updated connected component CC$[i, T]$ for interval $[m, i]$ (line 7). Each connected component CC$[i, T]$ corresponds to a maximal temporal motif. Then it invokes generateExpTM to obtain the set TF$[m, i]$ of maximal and non-expandable temporal motifs with interval $[m, i]$ (line 8). Finally, it returns the set TF of all the maximal and non-expandable temporal motifs.

E. Appendix: Detailed Examples of Algorithms FTM and DFTM

Example of algorithm FTM.

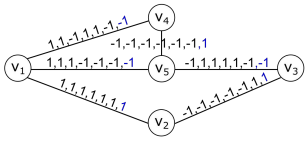
We consider the temporal graph G in Fig. 2a and frequency threshold $k = 4$. The corresponding TI-Table consists of 6 intervals in total, *i.e.,* $\{([1, 4], [1, 5], [1, 6]), ([2, 5], [2, 6]), ([3, 6])\}$.

(1) For row $m = 1$, computeRES returns R$[1, 4] = \emptyset$, R$[1, 5] = \{(v_2, v_3)\}$, R$[1, 6] = \{(v_1, v_2), (v_4, v_5)\}$, and EMaxIntvl$[(v_1, v_2), (v_4, v_5), (v_2, v_3)] = [[1, 6], [1, 6], [1, 5]]$. (a) For $i = 6$: generateMaxTM fetches edges from R$[1, 6]$, generates ccs $G_{s_1}(\{(v_1, v_2)\})$ and $G_{s_2}(\{(v_4, v_5)\})$ both with interval $[1, 6]$ (case (1)), and generateExpTM adds them to TF (H_4 and H_5 in Fig. 2c, respectively) as the beginning timestamp of $[1, 6]$ is equal to $m = 1$. (b) For $i = 5$: generateMaxTM adds edges from R$[1, 5]$ to G_{s_1}, and generates cc G_{s_3} with interval $[1, 5]$ (case (2)), which is added to TF with generateExpTM (H_6 in Fig. 2d) as $m = 1$. (c) For $i = 4$: the algorithm does nothing as R$[1, 4] = \emptyset$.

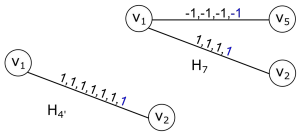
(2) For row $m = 2$, computeRES returns R$[2, 6] = \{(v_1, v_2), (v_4, v_5)\}$ and R$[2, 5] = \{(v_2, v_3), (v_3, v_5)\}$, and updates EMaxIntvl$[(v_3, v_5)] = [2, 5]$. (a) For $i = 6$: G_{s_1} and G_{s_2} with interval $[1, 6]$ are generated, which are left expandable ($1 < m = 2$). (b) For $i = 5$: generateMaxTM combines G_{s_1}, G_{s_2} and edges from R$[2, 5]$ to generate a cc G_{s_4}, and updates its interval with $[2, 5]$ (case (3)). As the beginning timestamp of $[2, 5]$ is $2 = m$, generateExpTM adds the cc G_{s_4} to TF (H_1 in Fig.2b).

(3) For row $m = 3$, computeRES returns R$[3, 6] = \{(v_1, v_2), (v_4, v_5)\}$ and keeps EMaxIntvl unchanged. G_{s_1} and G_{s_2} with interval $[1, 6]$ are generated again, which are left expandable ($1 < m = 3$).

Finally, algorithm FTM returns the set TF = $\{H_1, H_4, H_5, H_6\}$ of maximal and non-expandable temporal motifs.



(a) Updated temporal graph G' with interval $[1, 7]$



(b) Temporal motifs $H_{4'}$ and H_7 of G' with interval $[1, 7]$ and $[4, 7]$, respectively

Fig. 3: Running example for incremental algorithm

Example of algorithm DFTM**.** Consider the updated temporal graph G' in Fig. 3a of temporal graph G in Fig. 2a with $T = 6$ and $\Delta T = 1$, frequency threshold $k = 4$, and the set TF$^\circ$ of maximal and non-expandable temporal motifs for G computed by FTM in Example E. The updated TI-Table with $T + \Delta T = 7$ consists of 10 intervals, *i.e.,* $\{([1, 4], [1, 5], [1, 6], [1, 7]), ([2, 5], [2, 6], [2, 7]), ([3, 6], [3, 7]), ([4, 7])\}$.

(1) For row $m \in [1, 2]$, TF$[m, i]$ = TF$^\circ[m, i]$ for $i \in [m+3, 5]$, *i.e.,* the maximal and non-expandable temporal motifs keep unchanged for intervals $\{[1, 4], [1, 5], [2, 5]\}$.

(2) For row $m \in [1, 3]$ and $i \in [6, 7]$, sets TF$^\circ[m, 6]$ are used to compute edge sets R$[m, 6]$ and R$[m, 7]$. Then DFTM invokes procedures generateMaxTM and generateExpTM to obtain maximal and non-expandable temporal motifs $H_{4'}$ with interval $[1, 7]$ and H_5 with interval $[1, 6]$ shown in Figs. 3b

& 2c, respectively. Note that H_4 in $\mathsf{TF}^{\circ}[1,6]$ is right expandable after updates.

(3) For row $m = 4$, there is only one interval $[4, 7]$, and DFTM finds maximal and non-expandable temporal motif H_7 with interval $[4, 7]$ in Fig. 3b, which follows exactly the same way as FTM.

Finally, algorithm DFTM returns the set $\mathsf{TF} = \{H_1, H_{4'}, H_5, H_6, H_7\}$.

F. Appendix: correctness proofs of algorithm FTM

Proof of Proposition 5: We show this by a loop invariant.

Loop invariant: before the start of each row m (line 3), algorithm FTM finds all the maximal and non-expandable temporal motifs for all intervals in rows from 1 to $(m-1)$ in the TI-Table.

For row $m = 1$, it is easy to verify that the loop invariant holds. Assume that the loop invariant holds for row $m > 1$, and we show that the loop invariant holds for row $m+1$. (1) By Proposition 2, procedure generateMaxTM correctly generates all maximal temporal motifs for each interval at the $(m+1)$-th row in the TI-Table. The frequency threshold k is also assured from the arrangement of the TI-Table.

(2) By Proposition 3, procedure generateExpTM correctly checks whether a maximal temporal motif is left expandable or not, and generates maximal and non-expandable temporal motifs. This guarantees that FTM correctly finds the maximal and non-expandable temporal motifs for all intervals in the $(m+1)$-th row. This shows that the loop invariant holds for row $m+1$.

Putting these together, and we have the conclusion. □

G. Appendix: correctness proofs of algorithm DFTM

Proof of Proposition 6: We show this by proof by induction.

(1) For row $m = 1$, some edges of any temporal motif G_s in $\mathsf{TF}^{\circ}[1, T-1]$ must change their labels at the timestamp T, hence G_s is not right expandable. They are obviously not left expandable as $m = 1$. It is the same for $i \in [k, T-2]$. As sets $\mathsf{S}[1, T]$ and $\mathsf{R}[1, T-1]$ keep unchanged after update, the conclusion holds for $m = 1$.

(2) Assume that the conclusion holds for row $m > 1$, we then show that it holds for row $m+1$. For row $m+1$, similarly any temporal motif G_s in $\mathsf{TF}^{\circ}[m+1, T-1]$ is non-expandable, as the maximal and non-expandable temporal motifs keep unchanged for the rows above $m+1$. It is the same for $i \in [m+k, T-2]$. As sets $\mathsf{S}[m+1, T]$ and $\mathsf{R}[m+1, T-1]$ keep unchanged after update, the conclusion holds for $m+1$.

Putting these together, we have the conclusion. □

Proof of Proposition 7: This is proved by showing that for $m \in [1, T-k+1]$, if edge $e \in \mathsf{S}[m, T]$ and $e \notin \mathsf{TF}^{\circ}[m, T]$, then $e \notin \mathsf{TF}[m, i]$ for $i \geq T$.

(1) It is obvious that the edges in $\mathsf{TF}[m, i]$ must belong to $\mathsf{S}[m, T]$. (2) Assume that edge $e \in \mathsf{TF}[m, i]$ for $i \geq T$. As $e \in \mathsf{TF}[m, i]$, e belongs to a maximal and non-expandable motif in $\mathsf{TF}[m, i]$ that consists of edges $e, \cdots, e_h$ ($h \geq 0$). Then we know $e, \cdots, e_h \in \mathsf{S}[m, i] \subseteq \mathsf{S}[m, T]$. It implies that $e, \cdots, e_h$ belong to a maximal and non-expandable motif in $\mathsf{TF}^{\circ}[m, T]$. Therefore, edge $e \in \mathsf{TF}^{\circ}[m, T]$, which is a contradiction.

Putting these together, we have the conclusion. □

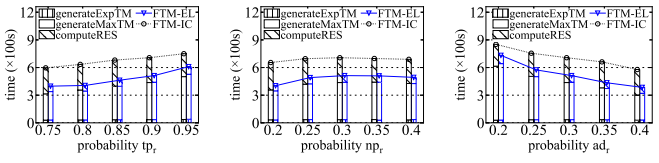
H. Appendix: detailed algorithm complexity analyses

(1) Time complexity of the procedure generateMaxTM. Let $|E_m|$ be the number of edges in temporal subgraph $G_s(\mathsf{S}[m, m+k-1])$. Note that the dynamic connectivity checking for an edge with the ccs in $\mathsf{CC}[i, T]$ ($i \in [m+k-1, T]$) utilizes disjoint sets [17]. For each e, cases (1) and (2) take $O(1)$ constant time, and case (3) takes $O(\min(|E_s|, |E_{s'}|))$ time, where $|E_s|$ and $|E_{s'}|$ are the numbers of edges in ccs G_s and $G_{s'}$, respectively. The worse case is that all edges in $\mathsf{S}[m, m+k-1]$ form a single connected component. This implies that cases (1), (2) and (3) all take $O(|E_m|)$ time. Therefore, it takes generateMaxTM $O(|E_m|)$ time to generate all the maximal temporal motifs for the intervals in the m-th row in the TI-Table.

(2) Time complexity of the procedure generateExpTM. Let $|E_m|$ be the number of edges in temporal subgraph $G_s(\mathsf{S}[m, m+k-1])$. In the worse case, each edge leads to a newly generated connected component. Hence, there are at most $|E_m|$ newly generated connected components for the m-th row. As the checking takes $O(1)$ constant time for each newly generated connected component, we know that procedure generateExpTM takes $O(|E_m|)$ time to generate all maximal and non-expandable temporal motifs for the intervals in the m-th row.

(3) Time complexity of the algorithm FTM. Let $|E_m|$ be the number of edges in temporal subgraph $G_s(\mathsf{S}[m, m+k-1])$, and let $|V_e|$ be the number of nodes in the IC tree of edge $e \in E$. We also denote $maxE_m$ as the largest number of edges in all temporal subgraphs $G_s(\mathsf{S}[m, m+k-1])$, $maxV_e$ as the largest number of nodes in all IC trees. Note that here $|E_m| \leq |E|$ for all $m \in [1, T-k+1]$ and $maxE_m$ is obviously smaller than $|E|$.

For each row m, (1) procedure computeRES takes $O(|E|)$ and $O(|E| \log maxV_e)$ time, respectively, when using the EL table and IC trees, (2) procedure generateMaxTM takes $O(|E_m|)$ time, and (3) procedure generateExpTM takes $O(|E_m|)$ time. There are in total $O(T-k+1) = O(T)$ rows in the TI-Table. Hence, algorithms FTM-EL and FTM-IC take $O(T|E|)$ and $O(T|E| \log maxV_e)$ time, respectively. Note that the motifs in the TF have $O(T^2)$ distinct intervals, and the motifs with the same interval contain $O(|maxE_m|)$ edges in total. Therefore, though we only need $O(T|E|)$ or $O(T|E| \log maxV_e)$ time to obtain TF, we take $O(T^2|maxE_m|) = O(T^2|E|)$ worst-case time to output all temporal motifs from TF.



(a) Vary tp_r (SYNDATA) (b) Vary np_r (SYNDATA) (c) Vary ad_r (SYNDATA)

Fig. 4: Algorithms FTM-EL vs. FTM-IC

(4) Space complexity of the algorithm FTM. The space cost of FTM is dominated by its key data structures. (1) For IC trees, they cost $O(maxV_e|E|)$ space, as each IC tree costs $O(maxV_e)$ space. (2) For the EL table, it costs $O(T|E|)$ space. (3) For array EMaxIntvl, it costs $O(|E|)$ space for all edges. (4) For connected components, they cost $O(|E|)$ space in total, as only one copy is maintained for all $\mathsf{CC}[i,T]$ ($i \in [m+k-1,T]$). (5) For the set TF of maximal and non-expandable temporal motifs, it can only cost $O(TmaxE_m)$ space. For each row m, we can save the motifs with the interval $[m,T]$ as all edges from ccs in $\mathsf{CC}[T,T]$ that need to be saved (*i.e.,* edges in $\mathsf{R}[m,T]$), but save the motifs with the interval $[m,i]$ as all the unsaved edges from newly generated ccs in $\mathsf{CC}[i,T]$, $i \in [m+k-1,T-1]$ that need to be saved (*i.e.,* edges in $\mathsf{R}[m,i']$, $i' \in [m+k-1,T]$) and the pointers to the motifs with already saved edges. As there are in total $O(maxE_m)$ edges in $\mathsf{S}[m,m+k-1]$, there are at most $O(maxE_m)$ pointers to distinct saved motifs. Therefore, TF only cost $O(maxE_m)$ space to save edges and pointers for each row.

From the above mentioned, FTM-EL and FTM-IC take $O(T|E| + TmaxE_m)$ and $O(maxV_e|E| + TmaxE_m)$ space, respectively.

(5) Time complexity of the algorithm DFTM. (a) For each row $m \in [1,T+k-1]$, procedure computeRES takes $O(|E_{m,TF}|)$ and $O(|E_{m,TF}|\log\ maxV_e)$ time when using the EL table and IC trees, respectively, and procedures generateMaxTM and generateExpTM take $O(|E_{m,TF}|)$ time. Hence, this part in total takes $O(TmaxE_{TF})$ and $O(T\ maxE_{TF}\log maxV_e)$ time, respectively, when using the EL table or IC trees. (b) For each row $m \geq T-k+2$, the time complexity is the same as the static algorithm. Hence, this part in total takes $O(\Delta T|E|)$ and $O(\Delta T|E|\log maxV_e)$ time, respectively, when using the EL table or IC trees.

Therefore, incremental algorithms DFTM-EL and DFTM-IC take $O(TmaxE_{TF} + \Delta T|E|)$ and $O((TmaxE_{TF} + \Delta T|E|)\log\ maxV_e)$ time, respectively.

I. Appendix: extra experimental tests

Exp-1.5. We varied tp_r from 0.75 to 0.95 while fixed $k = 10$, and used the entire temporal graphs for SYNDATA. Note that tp_r controls the probability of each activated edge that activates its copy in the next snapshot, which means that it controls the frequency of label changing for each edge. The result is reported in Fig. 4a.

When varying the probability tp_r, the running time of two algorithms increases with the increment of tp_r. The reason is that for larger probability tp_r, edges change their labels less frequently, and larger temporal motifs are generated due to larger $|E_m|$. Moreover, FTM-EL is 1.43 times faster than FTM-IC on average.

When varying the probability tp_r, all three procedures in FTM-EL, generateMaxTM and generateExpTM in FTM-IC increase with the increment of tp_r, as larger temporal motifs are generated. However, computeRES in FTM-IC decreases with the increment of tp_r, as IC trees have fewer nodes and are faster when edges change their labels less frequently. Further, generateMaxTM takes the most time which occupies 79% and 54% time in FTM-EL and FTM-IC on average, respectively.

Exp-1.6. To evaluate the impacts of probability np_r, we varied np_r from 0.2 to 0.4, and used the same setting as Exp-1.5. Note that np_r controls the probability of each activated edge that activates its neighboring edges in the next snapshot, which means that it has indirect influence on the frequency of label changing for edge. The result is reported in Fig. 4b.

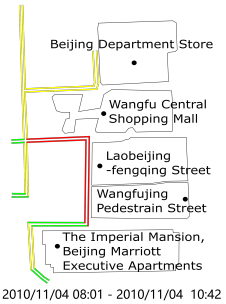
When varying the probability np_r, the running time of two algorithms increases with the increment of np_r when $np_r \leq 30$, and slightly decreases when np_r is larger. The reason is that for larger probability np_r, edges are more likely to activate their neighboring edges in the next snapshot, and help each activated neighbor edge to activate its copy in the next snapshot, which results in larger $|E_m|$ for each row m and more temporal motifs. However, when np_r is enough large (*i.e.,* $np_r > 30$), each edge exerts more influence on its neighboring edges in the next snapshot, which results in larger but less temporal motifs. Moreover, FTM-EL is 1.44 times faster than FTM-IC on average.

When varying the probability np_r, all three procedures in FTM-EL, generateMaxTM and generate- ExpTM in FTM-IC increase, and computeRES in FTM-IC decreases with the increment of np_r, along the same reason as Exp-1.5. Further, generateMaxTM takes the most time which occupies 79% and 55% time in FTM-EL and FTM-IC on average, respectively.

Exp-1.7. To evaluate the impacts of activation density ad_r, we varied ad_r from 0.2 to 0.4, and used the same setting as Exp-1.5. Note that ad_r controls the percentage of activated edges (labels are 1), which means it controls the proportion of different labels. The result is reported in Fig. 4c.

When varying the probability ad_r, the running time of two algorithms decreases with the increment of ad_r. The reason is that for larger percentage ad_r of activated edges, the generator needs to activate more edges in the first snapshot, which causes edges to change their labels more frequently and smaller temporal motifs to be generated due to more activated neighbor edges and decayed probabilities np_r and tp_r. Moreover, FTM-EL is 1.38 times faster than FTM-IC on average.

When varying the probability ad_r, all three procedures in FTM-EL, generateMaxTM and generate- ExpTM in FTM-IC decrease with the increment of ad_r, as smaller temporal motifs are generated. However, computeRES in FTM-IC increases with the increment of ad_r, as IC trees have more nodes and are slower when edge labels change more frequently. Further,



(a) Temporal motif with hybrid traffic condition labels at Beijing on BJDATA

Fig. 5: Case studies on real-life datasets

generateMaxTM takes the most time which occupies 79% and 57% time in FTM-EL and FTM-IC on average, respectively.

Case study: temporal motif with hybrid labels.

We use our temporal motifs to discover hybrid traffic condition patterns from the result of the algorithm FTM with $k = 10$. As Fig. 5a depicts, the motif has 15 nodes, 14 edges and the time interval length is 33. It corresponds to the traffic conditions in the *Wangfujing shopping area at Beijing* during time interval [08:01, 10:42]. It is the large area with unchanged traffic conditions for a long time involving with roads, *e.g.,* Chenguang Street, Datianshuijing Hu Tong, Wangfujing West Street, South Koudai Hu Tong, Daruanfu Hu Tong and Xiagongfu Street. Observe that there are two congested bottleneck roads near Wangfu Central Shopping Mall and Laobeijingfengqing Street in Fig.5a, which cause a traffic jam due to their popularity. Other roads with slow labels are influenced by the traffic jam. Note that, there are some roads with fast labels in the pattern in Fig.5a, which indicates potential solutions to improve the traffic conditions, such as arranging traffic policemen to guide vehicles to roads near The Imperial Mansion, Beijing Marriott Executive Apartments.

J. APPENDIX: DETAILED RELATED WORK

Network motifs in static networks. Network motifs, which mean the recurrent and statistically significant subgraphs or patterns of a large graph, have been applied to various fields, *e.g.,* biological networks [18]–[24], social networks [20], [21], [23]–[34], electrical circuits [35], software architectures [36], and transport networks [21], [23], [24], [37].

Most of these studies focus on efficient exact approaches to finding motifs with no more than five nodes [18], [23], [28], [29], [36], [38]. To handle the increasing needs of finding larger motifs in complex networks, subsequent studies begin to focus on motifs with more nodes in heterogenous graphs or uncertain graphs. [30]–[32] propose approximation approaches to counting larger motifs, based on color coding [30], [32], random work [30], [31] and adaptive sampling [32]. [24], [33], [34], [39] propose parallel approaches to counting larger motifs, based on Hadoop [33], [34], [39] and the TRUBA system [24]. [22], [27] focus on typed graphlets and colored graphlets to find motifs with node labels and edge labels in homogeneous graphs, respectively. Considering that there exists noisy and incomplete data, [19] and [21] propose sampling approaches to counting motifs in uncertain graphs.

These studies focus on network motifs in static networks, and do not consider temporal networks. In this study, we focus on analyzing network motifs in temporal networks.

Network motifs in temporal networks. Network motifs in temporal networks have attracted growing attentions. These studies re-define temporal motifs to meet various application demands, including social networks [40]–[50], communication networks [40], [41], [44], [48], [51]–[54], transaction networks [40]–[42], [44]–[47], [50], [55], transport networks [42], [43], [48] and biological networks [55], [56].

Some of these studies focus on temporal motifs with time-dependent edges. [57] firstly studies the dynamics of motifs in networks with time-dependent edges. Subsequent studies begin to define motifs with different time orders and different time windows on edges, *e.g.,* a partial time order [48], [51]–[54], a total time order [40]–[42], [45]–[47], [49], a local time window [49]–[54], [56], and a global time window [40]–[42], [45]–[49], [58]. A recent survey [59] systematically evaluates the impact of temporal inducedness and timing constraints (*i.e.,* local or global time windows) in some of these temporal motifs.

Other these studies focus on temporal motifs with each of node weights or edge weights (labels) displaying statics or similar dynamics over a user-defined period. [43], [55] define motifs with induced subgraphs and node weights evolving in a consistent trend over the time interval of a motif. [44] defines induced relational states (IRS) with edge labels and directions keeping constant over the time interval of induced subgraphs, which are somehow similar to our motifs. The difference between IRS and our model is that our motifs do not need to be induced subgraphs and have maximal and non-expandable constraints to reduce redundancy. Further, [44] uses subgraph enumeration to identify IRS in exponential time, while our motifs can be identified in PTIME. In conclusion, most these works focus on small motifs, and the discovery of these temporal motifs is computationally intractable. Further, no incremental solutions have been investigated to meet the need of the dynamic nature of temporal networks.

Similar but different concepts. There are also studies on graphs that bear similarities but are different from motifs, such as persistent connected components [60], lasting dense subgraphs [61], graph association rules [62], graph temporal association rules [63], dense temporal graphs [1], [3], [4], frequent graph patterns [64] and frequent pattern mining [65].

Persistent connected components defined in [60] focus on the node sets where nodes in the same set belong to the same connected component in consecutive snapshots. However, they do not model relationships among their edges, as nodes can be connected by other nodes which are not in node sets. Hence, persistent connected components do not fit the recurrence nature of motifs. Lasting dense subgraphs defined in [61] adopt the density scores of aggregate graphs from dense temporal

graphs, and are induced subgraphs with high density scores in the subset of graph snapshots. Graph association rules model association between entities in graphs [62], and graph temporal association rules model the association between events in temporal graphs [63]. Dense temporal graphs are temporal subgraphs with high density scores, which depend on the problems and applications, including the cohesive density of aggregate graphs and average degrees or weights of edges and nodes [1], [3], [4]. Frequent patterns, including frequent graph patterns, are well known to model relationships among the items or graphs in a database, whose methods determine all patterns that are present in at least a fraction of the transactions or the collection of graphs [64], [65].

The concepts of temporal graphs and subgraphs that we use essentially follow [3], [4], which were used to study the dense temporal graphs problem. Slightly different from their definition, we use edge labels, instead of edge weights, to represent the dynamic properties of temporal graphs.

REFERENCES

[1] P. Bogdanov, M. Mongiovì, and A. K. Singh, "Mining heavy subgraphs in time-evolving networks," in *ICDM*, 2011.

[2] M. Mongiovì, P. Bogdanov, and A. K. Singh, "Mining evolving network processes," in *ICDM*, 2013.

[3] S. Ma, R. Hu, L. Wang, X. Lin, and J. Huai, "An efficient approach to finding dense temporal subgraphs," *TKDE*, vol. 32, no. 4, pp. 645–658, 2020.

[4] S. Ma, R. Hu, L. Wang, X. Lin, and J. Huai, "Fast computation of dense temporal subgraphs," in *ICDE*, 2017.

[5] Y. Zheng, L. Capra, O. Wolfson, and H. Yang, "Urban computing: Concepts, methodologies, and applications," *ACM TIST*, vol. 5, no. 3, pp. 38:1–38:55, 2014.

[6] H. Wu, J. Cheng, S. Huang, Y. Ke, Y. Lu, and Y. Xu, "Path problems in temporal graphs," *PVLDB*, vol. 7, no. 9, pp. 721–732, 2014.

[7] L. Foschini, J. Hershberger, and S. Suri, "On the complexity of time-dependent shortest paths," *Algorithmica*, vol. 68, no. 4, pp. 1075–1097, 2014.

[8] C. Li, W. Yue, G. Mao, and Z. Xu, "Congestion propagation based bottleneck identification in urban road networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 5, pp. 4827–4841, 2020.

[9] H. Nguyen, W. Liu, and F. Chen, "Discovering congestion propagation patterns in spatio-temporal traffic data," *IEEE Transactions on Big Data*, vol. 3, no. 2, pp. 169–180, 2016.

[10] N. Hutcheon and J. W. Bialek, "Updated and validated power flow model of the main continental european transmission network," in *2013 IEEE Grenoble Conference*, 2013.

[11] T. V. Jensen and P. Pinson, "Re-europe, a large-scale dataset for modeling a highly renewable european electricity system," *Scientific data*, vol. 4, pp. 170 175:1–170 175:18, 2017.

[12] S. Pfenninger and I. Staffell, "Long-term patterns of european pv output using 30 years of validated hourly reanalysis and satellite data," *Energy*, vol. 114, pp. 1251–1265, 2016.

[13] S. Rozalis, E. Morin, Y. Yair, and C. Price, "Flash flood prediction using an uncalibrated hydrological model and radar rainfall data in a mediterranean watershed under changing hydrological conditions," *Journal of hydrology*, vol. 394, no. 1-2, pp. 245–255, 2010.

[14] V. Seal, A. Raha, S. Maity, S. K. Mitra, A. Mukherjee, and M. K. Naskar, "A real time multivariate robust regression based flood prediction model using polynomial approximation for wireless sensor network based flood forecasting systems," in *International Conference on Computer Science and Information Technology*, 2012.

[15] N. Ahmad, M. Hussain, N. Riaz, F. Subhani, S. Haider, K. S. Alamgir, and F. Shinwari, "Flood prediction and disaster risk analysis using gis based wireless sensor networks, a review," *Journal of Basic and Applied Scientific Research*, vol. 3, no. 8, pp. 632–643, 2013.

[16] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. Cambridge, Mass: MIT press, 2009.

[17] R. E. Tarjan, "Efficiency of a good but not linear set union algorithm," *Journal of the ACM*, vol. 22, no. 2, pp. 215–225, 1975.

[18] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, "Network motifs: simple building blocks of complex networks," *Science*, vol. 298, no. 5594, pp. 824–827, 2002.

[19] N. H. Tran, K. P. Choi, and L. Zhang, "Counting motifs in the human interactome," *Nature communications*, vol. 4, no. 1, pp. 2241:1–2241:8, 2013.

[20] X. Li, R. Cheng, K. C.-C. Chang, C. Shan, C. Ma, and H. Cao, "On analyzing graphs with motif-paths," *PVLDB*, vol. 14, no. 6, pp. 1111–1123, 2021.

[21] C. Ma, R. Cheng, L. V. Lakshmanan, T. Grubenmann, Y. Fang, and X. Li, "Linc: a motif counting algorithm for uncertain graphs," *PVLDB*, vol. 13, no. 2, pp. 155–168, 2019.

[22] S. Gu, J. Johnson, F. E. Faisal, and T. Milenković, "From homogeneous to heterogeneous network alignment via colored graphlets," *Scientific reports*, vol. 8, no. 1, pp. 12 524:1–12 524:16, 2018.

[23] N. K. Ahmed, J. Neville, R. A. Rossi, and N. Duffield, "Efficient graphlet counting for large networks," in *ICDM*, 2015.

[24] E. R. Ateskan, K. Erciyes, and M. E. Dalkilic, "Parallelization of network motif discovery using star contraction," *Parallel Computing*, vol. 101, pp. 102 734:1–102 734:12, 2021.

[25] P. Li, H. Dau, G. Puleo, and O. Milenkovic, "Motif clustering and overlapping clustering for social network analysis," in *INFOCOM*, 2017.

[26] H. Yin, A. R. Benson, J. Leskovec, and D. F. Gleich, "Local higher-order graph clustering," in *SIGKDD*, 2017.

[27] R. A. Rossi, N. K. Ahmed, A. Carranza, D. Arbour, A. Rao, S. Kim, and E. Koh, "Heterogeneous graphlets," *TKDD*, vol. 15, no. 1, pp. 9:1–9:43, 2020.

[28] X. Hu, Y. Tao, and C.-W. Chung, "Massive graph triangulation," in *SIGMOD*, 2013.

[29] A. Pinar, C. Seshadhri, and V. Vishal, "Escape: Efficiently counting all 5-vertex subgraphs," in *WWW*, 2017.

[30] M. Bressan, F. Chierichetti, R. Kumar, S. Leucci, and A. Panconesi, "Counting graphlets: Space vs time," in *WSDM*, 2017.

[31] X. Chen, Y. Li, P. Wang, and J. C. Lui, "A general framework for estimating graphlet statistics via random walk," *PVLDB*, vol. 10, no. 3, pp. 253–264, 2016.

[32] M. Bressan, S. Leucci, and A. Panconesi, "Motivo: fast motif counting via succinct color coding and adaptive sampling," *PVLDB*, vol. 12, no. 11, pp. 1651–1663, 2019.

[33] Z. Zhao, G. Wang, A. R. Butt, M. Khan, V. A. Kumar, and M. V. Marathe, "Sahad: Subgraph analysis in massive networks using hadoop," in *IPDPS*, 2012.

[34] L. Lai, L. Qin, X. Lin, and L. Chang, "Scalable subgraph enumeration in mapreduce: a cost-oriented approach," *VLDB Journal*, vol. 26, no. 3, pp. 421–446, 2017.

[35] S. Itzkovitz, R. Levitt, N. Kashtan, R. Milo, M. Itzkovitz, and U. Alon, "Coarse-graining and self-dissimilarity of complex networks," *Physical Review E*, vol. 71, no. 1, pp. 016 127:1–016 127:10, 2005.

[36] S. Valverde and R. V. Solé, "Network motifs in computational graphs: A case study in software architecture," *Physical Review E*, vol. 72, no. 2, pp. 026 107:1–026 107:8, 2005.

[37] P. Kaluza, A. Kölzsch, M. T. Gastner, and B. Blasius, "The complex network of global cargo ship movements," *Journal of the Royal Society Interface*, vol. 7, no. 48, pp. 1093–1103, 2010.

[38] A. Kara, H. Q. Ngo, M. Nikolic, D. Olteanu, and H. Zhang, "Counting triangles under updates in worst-case optimal time," in *ICDT*, 2019.

[39] F. N. Afrati, D. Fotakis, and J. D. Ullman, "Enumerating subgraph instances using map-reduce," in *ICDE*, 2013.

[40] A. Paranjape, A. R. Benson, and J. Leskovec, "Motifs in temporal networks," in *WSDM*, 2017.

[41] P. Liu, A. R. Benson, and M. Charikar, "Sampling methods for counting temporal motifs," in *WSDM*, 2019.

[42] C. Kosyfaki, N. Mamoulis, E. Pitoura, and P. Tsaparas, "Flow motifs in interaction networks," in *EDBT*, 2019.

[43] E. Desmier, M. Plantevit, C. Robardet, and J.-F. Boulicaut, "Trend mining in dynamic attributed graphs," in *ECML/PKDD*, 2013.

[44] R. Ahmed and G. Karypis, "Algorithms for mining the evolution of conserved relational states in dynamic networks," *KAIS*, vol. 33, no. 3, pp. 603–630, 2012.

[45] J. Wang, Y. Wang, W. Jiang, Y. Li, and K.-L. Tan, "Efficient sampling algorithms for approximate temporal motif counting," in *CIKM*, 2020.

[46] I. Sarpe and F. Vandin, "oden: Simultaneous approximation of multiple motif counts in large temporal networks," in *CIKM*, 2021.
[47] I. Sarpe and F. Vandin, "Presto: Simple and scalable sampling techniques for the rigorous approximation of temporal motif counts," in *SDM*, 2021.
[48] C. Song, T. Ge, C. Chen, and J. Wang, "Event pattern matching over graph streams," *PVLDB*, vol. 8, no. 4, pp. 413–424, 2014.
[49] P. Liu, N. Masuda, T. Kito, and A. E. Sarıyüce, "Temporal motifs in patent opposition and collaboration networks," *Scientific reports*, vol. 12, no. 1, pp. 1917:1–1917:11, 2022.
[50] N. Pashanasangi and C. Seshadhri, "Faster and generalized temporal triangle counting, via degeneracy ordering," in *SIGKDD*, 2021.
[51] L. Kovanen, M. Karsai, K. Kaski, J. Kertész, and J. Saramäki, "Temporal motifs in time-dependent networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2011, no. 11, pp. P11 005:1–P11 005:18, 2011.
[52] L. Kovanen, K. Kaski, J. Kertész, and J. Saramäki, "Temporal motifs reveal homophily, gender-specific patterns, and group talk in call sequences," *PNAS*, vol. 110, no. 45, pp. 18 070–18 075, 2013.
[53] Q. Zhao, Y. Tian, Q. He, N. Oliver, R. Jin, and W.-C. Lee, "Communication motifs: a tool to characterize social communications," in *CIKM*, 2010.
[54] S. Gurukar, S. Ranu, and B. Ravindran, "Commit: A scalable approach to mining communication motifs from dynamic networks," in *SIGMOD*, 2015.
[55] R. Jin, S. McCallen, and E. Almaas, "Trend motif: A graph mining approach for analysis of dynamic complex networks," in *ICDM*, 2007.
[56] Y. Hulovatyy, H. Chen, and T. Milenković, "Exploring the structure and function of temporal networks with dynamic graphlets," *Bioinformatics*, vol. 31, no. 12, pp. i171–i180, 2015.
[57] D. Braha and Y. Bar-Yam, "Time-dependent complex networks: Dynamic centrality, dynamic motifs, and cycles of social interactions," in *Adaptive Networks*, T. Gross and H. Sayama, Eds. Berlin, Heidelberg: Springer, 2009, pp. 39–50.
[58] Y. Li, Z. Lou, Y. Shi, and J. Han, "Temporal motifs in heterogeneous information networks," in *MLG Workshop@ KDD*, 2018.
[59] P. Liu, V. Guarrasi, and A. E. Sariyuce, "Temporal network motifs: Models, limitations, evaluation," *TKDE*, 2021, to be published.
[60] M. Vernet, y. Pigne, and E. Sanlaville, "A Study of Connectivity on Dynamic Graphs: Computing Persistent Connected Components," *4OR: A Quarterly Journal of Operations Research*, 2022, to be published.
[61] K. Semertzidis, E. Pitoura, E. Terzi, and P. Tsaparas, "Finding lasting dense subgraphs," *Data Mining and Knowledge Discovery*, vol. 33, no. 5, pp. 1417–1445, 2019.
[62] W. Fan, X. Wang, Y. Wu, and J. Xu, "Association rules with graph patterns," *PVLDB*, vol. 8, no. 12, pp. 1502–1513, 2015.
[63] M. H. Namaki, Y. Wu, Q. Song, P. Lin, and T. Ge, "Discovering graph temporal association rules," in *CIKM*, 2017.
[64] H. Cheng, X. Yan, and J. Han, "Mining graph patterns," in *Frequent Pattern Mining*, C. C. Aggarwal and J. Han, Eds. Cham: Springer, 2014, pp. 307–338.
[65] C. C. Aggarwal and J. Han, Eds., *Frequent Pattern Mining*. Cham: Springer, 2014.