Response to Reviewers

Shuwen Zheng, Chaokun Wang, Cheng Wu, Yunkai Lou, Hao Feng, Xuran Yang
School of Software, Tsinghua University, Beijing 100084, China
Email: {zsw22, wuc22, louyk18, fh20, yangxr22}@mails.tsinghua.edu.cn, chaokun@tsinghua.edu.cn

We would like to thank the meta-reviewer and reviewers for their considerable insights and valuable comments to our ICDE 2024 submission (926), which have been taken into careful consideration in our revision preparation. We believe that this response sufficiently addresses all issues raised by the reviewers. In our revised paper and response, all the modifications are colored in blue.

I. Response to Meta-reviewer

Comment: Please carefully address reviewers' detailed comments and the required changes in Question 12. Please refer to the required changes in Question 12.
Response: We appreciate your kind suggestion! We have carefully considered the detailed comments from the reviewers and revised the paper accordingly. Specifically, we answer Question 12 for Reviewer #1-#3, and resolve other problems as well. Note that all the modifications are marked in blue in the revised manuscript.

II. Response to Reviewer #1

A. R1-O1

Comment: The paper introduces a novel concept of time-bound community to explain the occurrence and extinction of communities in temporal graphs. However, the definition of this community appears intuitive, since the authors fail to provide any explanation for the rationale behind this specific definition. Additionally, it is encouraged for the authors to demonstrate the effectiveness of this community within the dataset, illustrating the properties of time-bound communities.
Response: We appreciate your valuable comment. We added explanation for the rationale of the definition of time-bound communities. Additionally, we demonstrated this definition within static and temporal real datasets. The details are as follow:

1. Definition explanation. **In order to facilitate comprehensible explanation for the rationale behind the time-bound community definition**, we concretized the evaluation f into calculating the number of edges within the set as follow:

> **Defnition 5** (Time-bound Community)**.** *Given a temporal graph $G = (V, E)$, a node set $V' \subseteq V$, and a time window w, if $\frac{|\check{E}_w(V')|}{|w|} > \frac{|\check{E}_{\overline{w}}(V')|}{|\overline{w}|}$, $\frac{2|\check{E}_w(V')|}{|V'|\cdot(|V'|-1)} > \frac{|\hat{E}_w(V')|}{|V'|\cdot|\overline{V'}|}$, and the (V', w)-time-bound structure $G' = (V', \check{E}_w(V'))$ is connected, then G' is said to be a time-bound community on G with w being its time window.*

The rationale behind this definition is to calculate the temporal and topological cohesiveness by comparing the density of the number of edges in terms of active duration and member nodes of a potential time-bound structure, respectively. For the temporal cohesiveness, the first inequality constraint $\frac{|\check{E}_w(V')|}{|w|} > \frac{|\check{E}_{\overline{w}}(V')|}{|\overline{w}|}$ demands that a time-bound community should possess more edges per unit time inside its time window w than outside w. For the topological cohesiveness, we considered to calculate the maximum possible number of edges with respect to the number of nodes in the simple graph, which equals $\frac{|V'|\cdot(|V'|-1)}{2}$ within V', and $|V'| \cdot |\overline{V'}|$ between V' and $\overline{V'}$. Then comparing the ratio of the actual number of edges versus the maximum possible number of edges, we go to the second inequality constraint $\frac{2|\check{E}_w(V')|}{|V'|\cdot(|V'|-1)} > \frac{|\hat{E}_w(V')|}{|V'|\cdot|\overline{V'}|}$, which requires that within the time window w the density of intra-community edges should be greater that of inter-community edges.

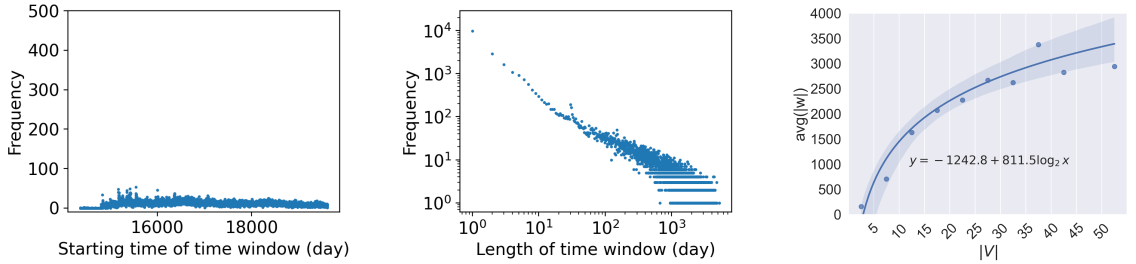
Additionally, this definition is expressive enough to cover a normal static community. Regarding the time window of the static community boundless, i.e., $|w| \to +\infty$ and $|\overline{w}| \to 0$, the first inequality constraint can be naturally satisfied. The second constraint can be written as $\frac{2|\check{E}(V')|}{|V'|\cdot(|V'|-1)} > \frac{|\hat{E}(V')|}{|V'|\cdot|\overline{V'}|}$, which demands the intra edges of V' should be denser than inter edges of V'. Therefore, the time-bound definition is an extension of intuitive static communities.

In Section II of the revised manuscript, we added the following descriptions to explain the rationale:

> Definition 5 imposes two inequality constraints for a time-bound community. The first one ensures temporal cohesiveness by requiring higher edge density inside the time window w than outside. The second enforces topological cohesiveness by demanding greater edge density within V' compared to between V' and $\overline{V'}$. After verification, a majority of communities in the real datasets align with our definition.

2. Effectiveness demonstration. **To demonstrate the effectiveness of this community definition within datasets**, its ability to characterize the communities in the temporal datasets, which had been collected from different online platforms, was scrutinized.

Specifically, we examine if groups in MathoverFlow, ServerFault, AppleSX, PhiloSX are time-bound communities. With the member nodes being the node set V' and the discussion period being the time window w, all of the groups comply with the topological constraint, and a minority of them (75.5%, 67.7%, 84.4%, and 73.4% in the four datasets respectively) adhere to the temporal constraint.



(a) Distribution of the start times (b) Distribution of the window lengths (c) Correlation of the window length and the community size

Fig. 1: Features of time windows in AppleSX circumscribing communities that align with the definition.

Non-compliance of the first constraint by certain groups could be attributed to the common overlapping amongst user groups. The users in V' have the possibility to engage in additional activities outside of the time window w, resulting in a significant ratio of $\frac{|\tilde{E}_{\overline{w}}(V')|}{|\overline{w}|}$. Time-bound communities should not include user groups that take part in an activity during a specific interval and actively participate in other activities in some neighboring time periods. Therefore, the rationality of our proposed community definition is upheld.

Ruling out a minority of communities that are not compliant with the definition, the features maintains a close similarity to the previous results. Figure 1 illustrates that Upon filtering for the communities in AppleSX that meet the definition criteria, the statistical features of their corresponding time windows exhibit close resemblance to the overall distribution of time windows (which is reported in our paper). Specifically, both the starting time and the length of the chosen time windows continue to present a uniform distribution and a power-law distribution respectively, while a manifest logarithmic correlation persists between the window length and community size.

Additionally, we examined the Email-Eu-core network[1], a static dataset of email communications from a large European research institution. Denoting the nodes of the community by V', we found that all communities in Email-Eu-core satisfy $\frac{2|\tilde{E}(V')|}{|V'|\cdot(|V'|-1)} > \frac{|\hat{E}(V')|}{|V'|\cdot|\overline{V'}|}$. This finding serves as empirical evidence for our hypothesis: the time-bound community definition is readily adaptable to conventional static communities.

B. R1-O2

Comment: The authors employed an index-based approach to accelerate the generation of temporal edges, while it appears to be a trivial extension from the baseline FastSGG. The authors are encouraged to highlight the differences and contributions of TED index algorithm compared to the baseline, as well as the challenges with respect to generating temporal graphs.

Response: Thanks for your comment. We underlined the differences and contributions of our GTB compared to the baseline FastSGG. We also emphasized that GTB addresses the challenges of the temporal graph generation——a task at which FastSGG does not excel. Our experimental results demonstrate the superior of GTB over FastSGG. The details are as follows:

1. Challenges of temporal graph generation. We **highlight the challenges with respect to generating temporal graphs** from the following standpoints:

Firstly, challenges are encountered in characterizing communities susceptible to extinction. Most existing temporal graph generators either overlooks community extinction, or simplistically model it by a single possibility of the departure of member nodes from the community at each timestamp. However, real communities may display different styles of formation and extinction. In our work, we adopt a stratagem of time boundary limitation, leaving user-directed configurability of extinction styles via different distribution settings. Also, the exploration of common features of time windows across different real temporal dataset presents a considerable hurdle.

Secondly, maintaining the configurability of a temporal graph generator for users is challenging. Real-world networks usually display rich degree distribution patterns, including power-law, exponential and log-normal distributions, which is overlooked by some temporal graph generation methods such as TESNG-M and RTGEN++. Additionally, communities may exhibit diverse paces of formation and extinction, corresponding to different timestamp distributions, whereas existing temporal graph generators such as STM and EpiCNet tend to capture graph evolution by several fixed possibilities per snapshot, confined to only exponential distributions. Therefore, it is vital but challenging to improve the configurability of a temporal graph generator to mimic user-given distributions.

The third challenge lies in the pursuit of efficiency in generating temporal graphs with perishable communities. The swift generation of large temporal datasets is essential for practical application. However, existing temporal graph generators tend to function on an incremental snapshot basis, leading to excessive time consumption, especially for largely spanning temporal graphs. Furthermore, current graph generators tend to approach each community's generation as individual tasks. In our study, we strive to circumvent the prevalent snapshot-by-snapshot iteration in current methodologies and to investigate the potential for index structure transferability across different time-bound communities.

In Section I of the revised manuscript, we concisely pre-

[1]https://snap.stanford.edu/data/email-Eu-core.html

sented the pre-mentioned challenges:

However, three major challenges present themselves:
Characterization of communities susceptible to extinction. Most existing temporal graph generators either disregard community extinction, or simplistically model it by a member departure possibility per snapshot. The characterization of perishable communities continues to be a pending issue.
Configurability for user-defined distributions. Some methods, such as EpiCNet [25], fails to accommodate user-specified distributions. Yet, networks may exhibit distinct distributions [41]-[44] including timestamp patterns due to their diverse paces of formation and decay. Thus, to mirror real-world scenarios, integrating user-defined distributions into graph generation methods is essential but challenging.
Efficiency in generating graphs with perishable communities. Existing temporal graph generators, typically operating snapshot by snapshot and approaching each community's generation as separate tasks, are inefficient for large timespan and scale settings. Developing an efficient generator addressing these issues is therefore paramount for data requirements.

2. Differences and contributions. We **highlight the differences and contributions of our TED index algorithm GTB compared to the baseline FastSGG** from the following angles:

Firstly, GTB targets **the dynamic nature of graphs**, a feature not catered for by FastSGG which is tailored exclusively towards static graph scenarios. The dynamic aspects in GTB are twofold: at the community level, it allows for the imposition of an active time window constraint on communities that approaches real-world patterns; whereas, at the edge level, it facilitates the modeling and generation of temporal edges adhered to the given timestamp distribution via the TED model.

Additionally, GTB maintains **configurability for user-defined distributions** including the timestamp distribution, which is not considered by FastSGG and other mainstream graph generators. The TED model in GTB deconstructs temporal edges into three fundamental elements, namely: the source node, the target node, and the associated timestamp. Employing TED indexes, the TED model enables the generation of temporal edges conformed to the given distributions of the aforementioned elements.

Furthermore, GTB achieves a distinguished efficiency, giving the credit to the **transferable TED index**. The construction of index structures requires one-time execution for the largest community in terms of degrees, and similarly, a singular build-out for the most enduring community in terms of timestamps. The transferability feature empowers the TED index to extend its utility to other time-bound communities, presenting a notable advantage over FastSGG, which mandates individual generation of different communities. As a result, when examining space complexity, FastSGG exhibits a requirement of $O(kn_v)$, with k and n_v respectively denoting the average in-degree and the total number of nodes in the graph. In stark contrast, GTB attains a more efficient space complexity $O(|V^*|+|w^*|)$, where V^* denotes the nodes of the largest time-bound community, and w^* corresponds to the time window of the longest-lived time-bound community. Using the TED index, it only costs $O(1)$ time for the TED model to generate a single temporal edge conformed to user-given distributions.

The experimental results indicate that GTB surpasses traditional graph generation methods including FastSGG, demonstrating enhanced efficiency and effectiveness. For time efficiency, excluding the time for file input/output operations, GTB can generate suitable datasets with ten billion temporal edges within one minute on our Ubuntu 16.04 server. Moreover, while controlling the number of communities to fluctuate, GTB outpaces FastSGG by trimming 26% of the time on average. For space efficiency, the memory consumption of GTB amounts to only 1/8.3 of the memory that FastSGG requires. In terms of effectiveness, the comparison between graphs generated GTB and the real temporal datasets demonstrates that GTB aptly emulates dynamics intrinsic to real-world time-bound communities.

We revised the manuscript in Section I to clarify the difference between GTB and other mainstream graph generators including FastSGG:

Table 1 outlines several essential attributes of GTB compared with other prevailing graph generators. Distinct from other methods, GTB targets dynamic nature of communities, preserves configurability for user-given distributions, and leverages transferable indexes for enhanced efficiency.

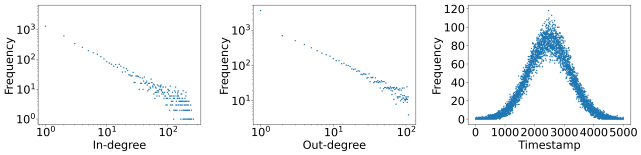
C. R1-O3

Comment: While the authors conducted experiments related to transferability, they only presented a distribution comparison between two distinct graphs. It is encouraged to provide a more experimental metric analysis, such as comparing statistical results between the transferred generated graph and exact calculated graph. Additionally, the authors should explore the transferability's performance under different node size conditions.

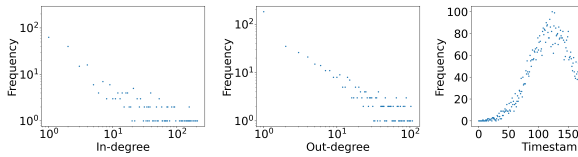
Response: Thanks for your comment. To enhance the experimental metric analysis with respect to transferability, we added experiments to present the statistical comparison of results between the community transferred and that exactly calculated, across different numbers of nodes. The details of the experiments are as follows:

1. Comparative graphs. **To compare results between the transferred generated graph and exact calculated graph**, we added the exact calculation of the small community the into comparison. Specifically, we considered the following three comparative objects:

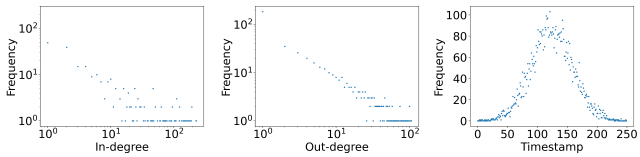
- An exact calculated large community that utilizes self-built indexes.
- **An exact calculated small community** that utilizes self-built indexes.



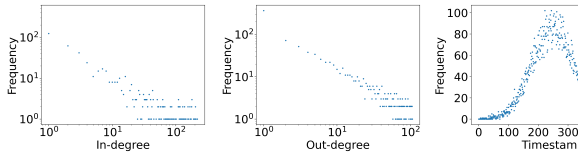
(a) The in-degrees, out-degrees and timestamps in the exact calculated large community (#nodes = 10000)



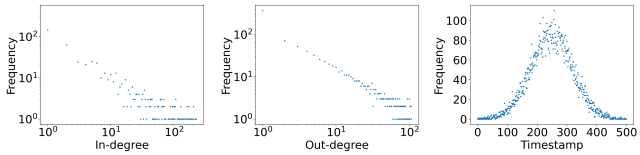
(b) The in-degrees, out-degrees and timestamps in the exact calculated small community (#nodes = 500)



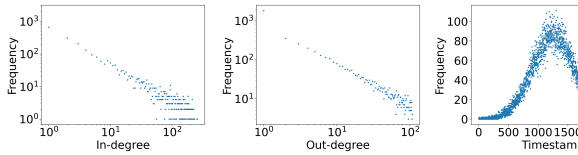
(c) The in-degrees, out-degrees and timestamps in the transferred small community (#nodes = 500)



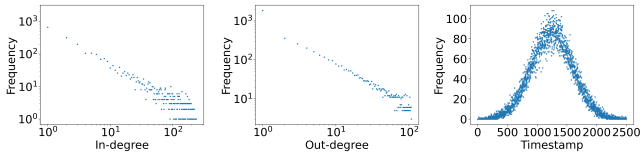
(d) The in-degrees, out-degrees and timestamps in the exact calculated small community (#nodes = 1000)



(e) The in-degrees, out-degrees and timestamps in the transferred small community (#nodes = 1000)



(f) The in-degrees, out-degrees and timestamps in the exact calculated small community (#nodes = 5000)



(g) The in-degrees, out-degrees and timestamps in the transferred small community (#nodes = 5000)

Fig. 2: Distribution plots of the exact calculated large communities, the exact calculated small communities, and the transferred small community under different node size conditions. The two exact calculated communities utilize individual tailored indexes, while the transferred community uses indexes transferred from the large community.

- A transferred small community that possesses with the same nodes as the exact calculated small community but utilizes indexes transferred from the large community.

In Section VI.E of the revised manuscript, we modified the descriptions as follow:

> To demonstrate the transferability, we focus on two exact communities (which use custom exact calculated indexes) with 10,000 nodes and 1,000 nodes respectively. Additionally, we examine a transferred community matching the exact small one but with indexes transferred from the large community.

Evaluating the transferability of index structures can be underscored by comparing statistical outcomes across the three comparative graphs.

2. Statistical metric. **To provide a more experimental metric analysis**, apart from directly comparing the distribution plots, we considered two additional frequently-used distribution distance metrics, i.e., Wasserstein distance (WD) and JS divergence (JSD). These metrics further measure the similarity of distributions of the exact calculated and transferred small communities from a more experimental perspective.

In Section VI.E of the revised manuscript, we added the following descriptions:

> Table I showcases low Wasserstein distances and JS divergences for the exact and transferred small communities under different conditions of the number of nodes, confirming the transferability of the TED model in GTB.

3. Different node size conditions. **To explore the transferability's performance under different node size conditions**, we conducted the experiment while setting the node size to range from 500 to 5,000 in the small community respectively, in order to make the distributions more statistically significant. The node count in the large community is fixed to 10,000.

4. Experimental results. The distribution plots of the three pre-mentioned comparative objects under different node size conditions are presented in Figure 2. It is clearly that the transferred results, i.e., distributions in Figure 2c, Figure 2e and Figure 2g, all resemble the exact calculated results in the large community, i.e., distributions in Figure 2a. Simultaneously, these transferred results respectively remain nearly the

TABLE I: Wasserstein distances (WD) and JS divergences (JSD) between distributions of the exact calculated small community and those of the transferred small community with different node count specifications.

#nodes	in-degree distr.		out-degree distr.		timestamp distr.	
	WD	JSD	WD	JSD	WD	JSD
500	0.031	0.00059	0.070	0.00043	0.0083	0.00016
1000	0.0087	0.00024	0.052	0.00038	0.0078	0.000047
5000	0.0085	0.00019	0.013	0.00013	0.0079	0.0000040

same as the exact calculated results in the large community, i.e., distributions in Figure 2b, Figure 2d and Figure 2f. Such experimental findings confirm that index structures in the TED model can be seamlessly transferred from large time-bound communities to other small ones. Note that the revised manuscript only include the distribution plots when the node count of the small community is 1000 due to length limitations.

Table I illustrates the Wasserstein distances and JS divergences for distributions of the exact calculated small community and the tranferred small community under different node count conditions. The statistical values in Table I appear consistently low, regardless of the community size. Such experimental results further demonstrate the transferability of the TED model in GTB.

In Section VI.E of the revised manuscript, we added the following analyses:

> Figure 12 (i.e., Figure 2a, Figure 2d and Figure 2e in this response, which is a partially selected version of Figure 2 in this response for lack of space) presents the distribution plots of the three communities. The observable similarity among their distributions tells the proficiency of index transfer in mimicking distributions. Table V (i.e., Table I in the response) showcases low Wasserstein distances and JS divergences for the exact and transferred small communities under different conditions of the number of nodes, confirming the transferability of the TED model in GTB.

D. R1-O4

Comment: The authors should consider presenting additional experiments to provide a more comprehensive evaluation of the GTB algorithm. For instance, the memory usage of the algorithm should be showcased, since the authors have already analyzed space complexity. Furthermore, conducting ablation experiments on the impact of the index and transferability on efficiency would provide a more comprehensive understanding of the GTB algorithm.

Response: Thanks for your suggestion. To provide a more comprehensive evaluation of GTB, we added the experiment on memory usage to show the space efficiency of GTB. We also conducted ablation experiments with respect to the index usage and index transfer respectively to demonstrate the indispensability of components in GTB. The details of the experiments are as follows:

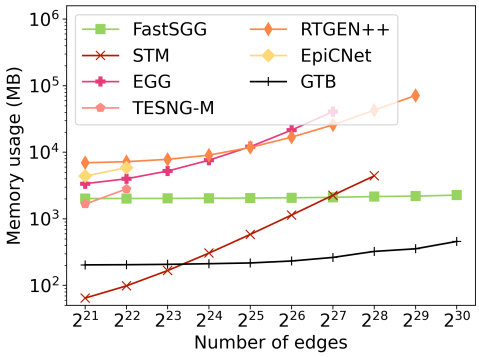


Fig. 3: Memory usage of GTB and other baseline methods varying edge count specifications.

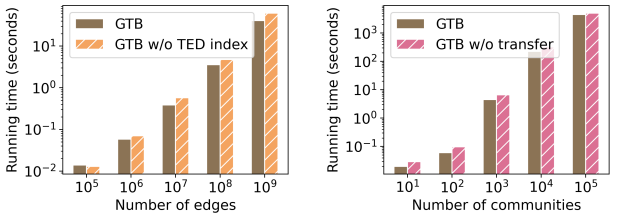
1. Memory usage of GTB. **To showcase the space efficiency of GTB**, we consider a suite of baseline methods including FastSGG, EGG, STM, TESNG-M, RTGEN++, and EpiCNet. We record the memory consumption of GTB alongside these baselines during the generation of graphs at different scales, as depicted in Figure 3. GTB beats other baseline methods especially when generating a large-scaled graph, owing to our well-designed index structures. It is noteworthy that GTB even keeps a greater spatial advantage than FastSGG while taking into account the temporal nature of graphs.

In Section VI.C of the revised manuscript, we added the following descriptions:

> We showcase GTB's space efficiency by detailing memory usage for generating graphs with 2^{21}-2^{30} edges. Figure 3 demonstrates that GTB spatially outperforms FastSGG and other temporal graph generation baselines, with community extinction considered.

2. Ablation study of GTB. **To demonstrate the impact of the index usage on efficiency of GTB**, we consider the variant GTB_{ID}, which avoids using indexes, instead directly queries against the approximate CDF to sample values from a given distribution. We incrementally vary the number of nodes and edges in the generation configurations to record the time consumption of GTB and GTB_{ID}, maintaining a constant count of communities. This is because the index establishment is more sensitive to fluctuations in the scale of communities rather than the number of communities in the generated graph. As Figure 4a shows, GTB outstrips GTB_{ID} by an average margin of 33%, underscoring the substantial gains in efficiency attributable to index employment.

To examine the impact of the index transfer on efficiency of GTB, we consider the variant GTB_{TR}, which avoids transferring indexes, instead constructs discrete indexes for individual time-bound communities. We record the running time of GTB and GTB_{TR} with increased community counts in the generation task, since the advantages of transferring indexes are closely related to the number of communities. As Figure 4b shows, GTB saves 13% of the time compared to GTB_{TR} on average. This is because GTB only builds indexes for the most extensive and the most enduring communities thus capable of cutting unnecessary consumption.



(a) The impact of index utilization (b) The impact of index transfer

Fig. 4: Ablation study for GTB.

In Section VI.G of the revised manuscript, we added the following descriptions:

> This section evaluates the impact of index usage and transfer on the operational efficiency, analyzing variant GTB_{ID} that bypasses indexing for direct CDF queries and variant GTB_{TR} that foregoes index transfer in favor of distinct indexes for each time-bound community.
> To ascertain the impact of indexing, Figure 14a (i.e., Figure 4a in this response) shows the execution time for GTB and GTB_{ID} to generate graphs with varied number of edges. GTB averages 33% faster than its counterpart, highlighting efficiency from using indexes.
> In a similar vein, Figure 14b (i.e., Figure 4b in this response) expounds on the influence of index transfer by charting the time expenditure of GTB and GTB_{TR} while generating graphs across community sizes. GTB saves 13% of the time compared to GTB_{TR} on average, owing to transferring indexes from mainly large and persistent communities. Such experimental results demonstrate the role of transferability in GTB's efficiency.

E. R1-O5

Comment: Some parts of the experimental discussion are redundant, for instance, the efficiency of EpiCNet is reiterated in section D.

Response: We appreciate your comment. We simplified the discussion about the efficiency of EpiCNet in Section VI.D of the revised paper as follow:

> For EpiCNet, its extinction simulation is disabled due to its inability to mimic power-law distribution.

Apart from that, we made the following textual simplifications:

(1) In Section III.B of the revised manuscript, we simplified the description for the start timestamp pattern of time windows:

> We examine the time window patterns for time-bound communities within the MATHOVERFLOW network. Figure 3a shows the start timestamps t_s (in days) are almost uniformly distributed, which aligns with expectations since users are free to choose when to begin a discussion.

(2) In Section III.B of the revised manuscript, we simplified the description for the correlation pattern of average timespan and community size:

> Moreover, the timespan of a time-bound community might be correlated with its size, thus influencing time window assignment post-grouping. To verify, we categorize each time-bound community by its size $|V|$ into 11 buckets: $\{5i \le |V| < 5(i+1) | i \in \mathbb{N}, i \le 9\} \cup \{|V| \ge 50\}$. The average length of time windows in each bucket is presented in Figure 4. Our analysis uncovers a logarithmic rise in average timespan with community size:
>
> $$\mathbb{E}(T) = b + a \log(|V|), \quad (1)$$
>
> where the constants $a = 766.5$ and $b = -1108.0$ determined via linear regression. This finding is in line with the intuition that larger-scale time-bound communities tend to last longer.

(3) In Section IV.C of the revised paper, we simplified the analysis for time efficiency examination varying the number of edges:

> To begin with, we incrementally raise the number of edges from 2^{21} to 2^{30} by adjusting the number snapshots and the average number of edges per snapshot, recording time costs of each algorithm in Figure 7a, where GTB displays remarkable superiority over other baselines. For one thing, GTB outperforms other temporal graph generators, notably 9000x faster than TESNG-M and 3470x speedier than EpiCNet. For another, GTB showcases a speed quite comparable to FastSGG, the fastest static graph generation baseline, and uniquely features community extinction.

(4) In Section IV.C of the revised paper, we simplified the time efficiency analysis when varying the number of communities as follows:

> Furthermore, comparing GTB's time efficiency with FastSGG over the numbers of communities from 10^1 to 10^5, the numbers of nodes from 10^2 to 10^6, and the numbers of edges 10^3 to 10^7, Figure 7b shows that GTB consistently outpaces FastSGG, trimming the generation time by more than 26% on average. The superiority of our model stems from the advanced transferable indexes in the TED model, building them solely for the largest community and the longest-lived one, unlike FastSGG's repetitive distribution calculations.

In Section IV.C of the revised paper, we simplified the time efficiency analysis when varying the number of snapshots as follows:

> In terms of the time efficiency across different numbers of snapshots, we set the timespan to range from 2^7 to 2^{13}, while holding the number of edges steady at 10^4 to exclude graph scale effect on time. We compare GTB with all the temporal graph generator baselines except STM due to its inability to directly specify the number of nodes. Figure 7c shows GTB's faster performance over the baselines, maintaining consistently low generation time regardless of snapshot number. Notably, TESNG-M never completes within 10^4 seconds, and RTGEN++ encounters exceptions at 2^{10} snapshot or more.

(5) In Section IV.D of the revised manuscript, we simplified the description for adapting FastSGG to generate temporal graphs:

> For TESNG-M, RTGEN++ and EpiCNet, timestamps are set to span from 0 to 9. FastSGG is adapted by assigning edges uniformly distributed timestamps in $[0, 9]$. Also, GTB is synchronized with the baseline methods by presenting the changes in modularity under 10 timestamps.

(6) In Section IV.D of the revised manuscript, we simplified the experimental analysis on Figure 8:

> Also, GTB is synchronized with the baseline methods by presenting the changes in modularity under 10 timestamps. Figure 8e reveals that the modularity of GTB's communities undergoes intense rise and fall, marking their formation and extinction, bound to specific time intervals. In contrast, TESNG-M, RTGEN++, and FastSGG show more stable modularity over time.

(7) In Section IV.D of the revised manuscript, we simplified our descriptions and analyses with respect to the comparison against the four collected real-world datasets:

> Figure 9 presents the properties of time windows associated with time-bound communities in each dataset. First-row results imply uniform distributions of actual time-bound community start times, which GTB simulates well. The second row reveals GTB's network has power-law distributed time window lengths, closely mirroring trends in real data. Third-row outcomes show that the size and lifespan of the time-bound communities created by GTB share a close correlation to those found in real-world datasets. These similarities manifests GTB's ability to replicate dynamics intrinsic to real-world time-bound communities.

(8) In Section IV.D of the revised manuscript, we simplified our analyses with respect to the compliance verification of the densification law:

> We further verify if our generated data abide by the densification law [todo], which posits that the number of edges should grow super-linearly in the number of nodes over time. As shown in Figure 10, the alignment of scatter points in a log-log scale indicates the compliance of our generation with the densification law.

(9) In Section IV.D of the revised manuscript, we simplified our analyses with respect to the comparison between real and generated data using static graph metrics:

> Additionally, we validate the soundness of our generated data using static graph metrics including clustering coefficient (clus. coef.), diameter (diam.), and average betweenness centrality (avg. betw. centr.). Regarding the PHILOSX network as a representative, Table IV reveals that GTB aligns with the properties of real data more closely compared to baseline methods. The results on other datasets are similar.

(10) In Section IV.E of the revised manuscript, we simplified our experimental description as follows:

TABLE II: Statistics of the real datasets.

dataset	#nodes	#temporal edges	#time-bound communities	timespan (days)
MATHOVERFLOW	40,387	912,663	115,912	4997
SERVERFAULT	182,023	1,463,966	249,739	5511
APPLESX	104,846	612,107	100,716	5471
PHILOSX	13,349	196,363	17,087	4534

> To demonstrate the configurability, we direct GTB to generate two unique graphs with different in-degree, out-degree and timestamp distributions, detailed in Figure 11a and Figure 11b, respectively. Figure 11a reveals a sample graph with power-law in-degree, normal out-degree, and log-normal timestamp distributions, while Figure 11b depicts another with log-normal in-degrees, power-law out-degrees, and exponential timestamps. These results illustrate GTB's capability to generate edges with user-defined distributions as expected for application across diverse use cases.

F. R1-O6

Comment: The paper contains some mistakes, such as:
(1) In example 5, it should be "0/4 * 4" instead of "0/4*3", as the complement set contains four nodes.
(2) The numerical representation in Table 2 lacks standard formatting; for instance, "1463,966" should be "1,463,966".
Response: Thanks for your suggestion. The first mistake that you pointed out was corrected as follows:

> **Example 1.** *In Figure 5b, given* $V' = \{v_1, v_2, v_3, v_4\}$, $w = [64, 75]$, $\frac{|\check{E}_w(V')|}{|w|} = \frac{10}{11} > \frac{0}{132} = \frac{|\check{E}_{\overline{w}}(V')|}{|\overline{w}|}$ *and* $\frac{2|\check{E}_w(V')|}{|V'|(|V'|-1)} = \frac{2\cdot 10}{4\cdot 3} > \frac{0}{4\cdot 4} = \frac{|\hat{E}_w(V')|}{|V'|\cdot|\overline{V'}|}$ *hold, and the subgraph* $G' = (V', \check{E}_w(V'))$ *(marked purple in Figure 5b) is connected. So* G' *is a time-bound community on* G*, with its time window* $w = [64, 75]$.

The second mistake was corrected as shown in Table II of this response. Apart from these two mistakes, we made the following corrections in the revised manuscript:
(3) We corrected the grammar mistake in the input texts of Algorithm 1 of the revised paper as follows:

> **Input:** The numbers of nodes, temporal edges and time-bound communities n_v, n_e and n_c, the distributions of out-degree, in-degree and timestamp π_o, π_i and π_t, the power-law distribution parameter λ for community size, the number of overlap relationships m between time-bound communities, the time-bound community overlapping parameter Ω, and the time-bound community messing parameter δ.

(4) We corrected the parameters passed of the Binding function in Algorithm 1 of the revised paper as follows:

$$W := \{w_1, \cdots, w_{n_c}\} \leftarrow \text{Binding}(n_c, \{|V_1|, \cdots, |V_n|\})$$

(5) We corrected the parameters passed of the Binding function in Algorithm 2 of the revised paper as follows:

Input: The number of time-bound communities n, and the sizes of each time-bound community $\{|V_1|, \cdots, |V_n|\}$.

(6) In Section III.C of the revised manuscript, we corrected the descriptions of Algorithm 3 as follows:

Ultimately, the algorithm returns temporal edges within and between each time-bound communities.

(7) In Section IV.A of the revised paper, we corrected the grammar mistake in the descriptions of Figure 5 as follows:

Figure 5 exemplifies out-degree indexing within the TED model, where the out-degree follows a power-law distribution with the exponent of -1.1, $(d_o)_{min} = 1$, and $(d_o)_{max} = 200$.

(8) In Section V of the revised manuscript, we corrected the grammar mistake in the proof of Lemma 2:

Constructing the probability functions of out-degree, in-degree, and timestamp respectively demands $O((d_o)_{max})$, $O((d_i)_{max})$, and $O(|w^*|)$ in both time and space complexities.

III. Response to Reviewer #2

We noticed that you indicated "n/a" to the source code availability. To assist with this, we have made our codes available at https://github.com/ukyotachibana0/GTB for your convenience.

A. R2-W1

Comment: Some figures could be enlarged in a higher resolution for easy seeing.

Response: Thanks for your suggestion. We increased the resolution and font size of several figures (e.g., Figure 2, Figure 5, and Figure 6) in the revised paper. We hope this modification facilitates easier reading.

B. R2-W2

Comment: More applications of temporal graph generation could be enhanced.

Response: We appreciate your suggestion. Graph generation has emerged as a critical technique to provide sufficient datasets for existing graph mining algorithms and further fuel subsequent methods.

Complex networks in the real world are often time-stamped and community-equipped, where the activeness of communities may associate to limited time periods. However, most systems depend on real-world temporal graphs for assessing the proposed methods, a practice that may fall short of fulfilling necessary scale criteria [1]. Consequently, there arises an imperative to generate becoming temporal datasets tailored for algorithms that operate on temporal graphs.

Specifically, the **applications of temporal graph generation** include:

- *Anti-money-laundering.* In the context of online financial transactions, criminals tend to transfer funds frequently within a short period to evade the regulations of money laundering prevention [2]. Thus, if several user accounts are detected to have frequent transactions in a short time, they may be considered suspicious, and further necessary surveillance actions can be taken. A series of algorithms that can screen these dubious groups on temporal graphs, such as [3]-[5], arise. Due to the privacy implications of financial data, temporal graph generators offering datasets with ground truths would greatly benefit the assessment of these algorithms.
- *Community question answering.* Community question answering (CQA) websites like Math Overflow facilitates the discussion of various questions. Users can submit questions, with opportunity for peers to engage by offering solutions and commenting on existing contributions. For one thing, the CQA websites need temporal information such as temporal interactive recordings to determine users' expertise and answers' correctness, which is seriously significant for recommendation purposes [6]-[8]. For another, community detection on CQA data contributes to many applications such as network monitoring [3] and group-based recommendations [9]. Employing synthetic temporal datasets could be conducive to evaluating the aforementioned CQA algorithms.
- *Temporal protein complexes detection.* In protein interaction networks (PINs), proteins often undergo a dynamic process of assembly and disassembly, corresponding to the formation and dissociation of protein complexes that are integral to performing distinct biological functions in the cellular processes [10]-[12]. In computational biology, the detection of temporal protein complexes plays an important role in exploring the dynamic modularization mechanisms of biological systems. A series of relevant algorithms have been proposed to address this problem [13]-[16]. However, the lack for large-scale available dynamic PIN data [17] hinders the development of relevant algorithms, thus necessitating the generation of temporal datasets with the ground truths of temporal protein complexes.

[1] M. Massri, Z. Miklos, P. Raipin, P. Meye, A. B. Pilet, and T. Hassan, "Rtgen++: A relative temporal graph generator," *Future Generation Computer Systems*, vol. 146, pp. 139–155, 2023.

[2] N. Heidarinia, A. Harounabadi, and M. Sadeghzadeh, "An intelligent anti-money laundering method for detecting risky users in the banking systems," *International Journal of Computer Applications*, vol. 97, no. 22, 2014.

[3] M. Zhong, J. Yang, Y. Zhu, T. Qian, M. Liu, and J. X. Yu, "A unified and scalable algorithm framework of user-defined temporal $(k, \mathcal{X})$-core query," *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[4] J. Yang, M. Zhong, Y. Zhu, T. Qian, M. Liu, and J. X. Yu, "Scalable time-range k-core query on temporal graphs," *Proceedings of the VLDB Endowment*, vol. 16, no. 5, pp. 1168–1180, 2023.

[5] C.-X. Zhu, L.-L. Lin, P.-P. Yuan, and H. Jin, "Discovering cohesive temporal subgraphs with temporal density aware exploration," *Journal of Computer Science and Technology*, vol. 37, no. 5, pp. 1068–1085, 2022.

[6] F. Wu, X. Duan, J. Xiao, Z. Zhao, S. Tang, Y. Zhang, and Y. Zhuang, "Temporal interaction and causal influence in community-based question answering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 10, pp. 2304–2317, 2017.

[7] L. Yang, M. Qiu, S. Gottipati, F. Zhu, J. Jiang, H. Sun, and Z. Chen, "Cqarank: jointly model topics and expertise in community question answering," in *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, 2013, pp. 99–108.

[8] V. Krishna and N. Antulov-Fantulin, "Temporal-weighted bipartite graph

model for sparse expert recommendation in community question answering," in *Proceedings of the 31st ACM Conference on User Modeling, Adaptation and Personalization*, 2023, pp. 156–163.
[9] P. Bedi and C. Sharma, "Community detection in social networks," *Wiley interdisciplinary reviews: Data mining and knowledge discovery*, vol. 6, no. 3, pp. 115–135, 2016.
[10] A.-C. Gavin, M. Bösche, R. Krause, P. Grandi, M. Marzioch, A. Bauer, J. Schultz, J. M. Rick, A.-M. Michon, C.-M. Cruciat et al., "Functional organization of the yeast proteome by systematic analysis of protein complexes," *Nature*, vol. 415, no. 6868, pp. 141–147, 2002.
[11] V. Spirin and L. A. Mirny, "Protein complexes and functional modules in molecular networks," *Proceedings of the national Academy of sciences*, vol. 100, no. 21, pp. 12 123–12 128, 2003.
[12] X. Li, M. Wu, C.-K. Kwoh, and S.-K. Ng, "Computational approaches for detecting protein complexes from protein interaction networks: a survey," *BMC genomics*, vol. 11, no. 1, pp. 1–19, 2010.
[13] R. R. Rani, D. Ramyachitra, and A. Brindhadevi, "Detection of dynamic protein complexes through markov clustering based on elephant herd optimization approach," *Scientific reports*, vol. 9, no. 1, p. 11106, 2019.
[14] X. Shen, L. Yi, X. Jiang, Y. Zhao, X. Hu, T. He, and J. Yang, "Neighbor affinity based algorithm for discovering temporal protein complex from dynamic ppi network," *Methods*, vol. 110, pp. 90–96, 2016.
[15] M. Li, X. Meng, R. Zheng, F.-X. Wu, Y. Li, Y. Pan, and J. Wang, "Identification of protein complexes by using a spatial and temporal active protein interaction network," *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 17, no. 3, pp. 817–827, 2017.
[16] L. Ou-Yang, D.-Q. Dai, X.-L. Li, M. Wu, X.-F. Zhang, and P. Yang, "Detecting temporal protein complexes from dynamic protein-protein interaction networks," *BMC bioinformatics*, vol. 15, pp. 1–14, 2014.
[17] E. D. Levy and J. B. Pereira-Leal, "Evolution and dynamics of protein interactions and networks," *Current opinion in structural biology*, vol. 18, no. 3, pp. 349–357, 2008.

In Section I of our revised manuscript, we added the following descriptions to enhance the applications of temporal graph generation:

> The discovery of such communities can be useful in many applications such as community detection on community question answering sites like Math Overflow for group-based recommendation [28] and network monitoring [29], money-laundering gang search on financial networks [30]-[32], and protein dynamic protein complex detection on temporal protein interaction networks [33]-[36].

IV. Response to Reviewer #3

We understand from your comment that there might have been a challenge accessing the source code. We have made the necessary codes available at https://github.com/ukyotachibana0/GTB for your ease of access.

A. R3-O1

Comment: The problem statement should be included in Section II, and the definition of the time-bound community needs refinement. According to Definition 4, a time-bound community can be disconnected?

Response: Thanks for your comment. We added the problem statement in Section II, and refined the definition of the time-bound community. The details are as follows:

1. Problem statement. To provide a formal and concise summary of the research problem, named as temporal graph generation featuring time-bound communities, we added **the following problem statement** in Section II of the revised manuscript:

> **Definition** (Temporal Graph Config)**.** A temporal graph config $\mathcal{C}$ can be denoted as $(n_v, n_e, n_c, \pi_o, \pi_i, \pi_t, \lambda, m, \Omega, \delta)$, where n_v, n_e and n_c are the target numbers of nodes, temporal edges and time-bound communities, π_o, π_i and π_t are the distributions of out-degree, in-degree and timestamp, λ is the power-law distribution parameter for community size, m is the number of overlap relationships between time-bound communities, Ω is the time-bound community messing parameter, and δ is the time-bound community messing parameter.
>
> **Problem** (Temporal Graph Generation Featuring Time-bound Communities)**.** *Given a temporal graph config $\mathcal{C}$, the goal of temporal graph generation featuring time-bound communities is to generate a temporal graph with n_v nodes, n_e temporal edges, n_c time-bound communities where the given distributions are followed.*

2. Definition refinement. We **refined the definition** by adding a connectivity constraint. Specifically, we updated Definition 5 in the Section II of the revised manuscript as follows:

> **Defnition 5** (Time-bound Community)**.** *Given a temporal graph $G = (V, E)$, a node set $V' \subseteq V$, and a time window w, if $\frac{|\check{E}_w(V')|}{|w|} > \frac{|\check{E}_{\overline{w}}(V')|}{|\overline{w}|}$, $\frac{2|\check{E}_w(V')|}{|V'|\cdot(|V'|-1)} > \frac{|\check{E}_w(V')|}{|V'|\cdot|\overline{V'}|}$, and the (V', w)-time-bound structure $G' = (V', \check{E}_w(V'))$ is connected, then G' is said to be a time-bound community on G with w being its time window.*

For coherence, modifications were incorporated into our GTB methodology to ensure the connectivity of the generated communities. Specifically, in the linking phase, GTB addresses each time-bound community by initially constructing a tree structure with respect to the member nodes, thus all members are connected. Apart from the n_v edges intrinsically generated, GTB proceeds to output the outstanding $n_e - n_v$ temporal edges, by querying TED indexes built based on given distributions as normal. Algorithm 1 (also updated in the revised manuscript) delineates the pseudocode pertinent to the aforementioned procedure. Note that a tree with n_v nodes characteristically contains $n_v - 1$ edges, leaving the requirements to generate $n_e - n_v + 1$ edges afterwards. Nevertheless, the integrity of our methodology is maintained, as misrecognizing the expected number of edges by one unit bears no material impact on the generation results from a statistical perspective.

In Section III.C of the revised manuscript, we added the following descriptions:

> Algorithm 3 (i.e., Algorithm 1 in the response) shows the process of the TED model to generate temporal edges. For each time-bound community, a tree structure is generated with respect to the community's nodes to assure connectivity (Line 4).

In Section IV.A of the revised manuscript, we updated the derivation and explanatory texts of the associated formula:

Algorithm 1 Linking

Input: The numbers of nodes and temporal edges n_v and n_e, the distributions of out-degree, in-degree and timestamp π_o, π_i and π_t, the nodes $\{V_1, \cdots, V_{n_c}\}$ and the time windows $\{w_1, \cdots, w_{n_c}\}$ of each time-bound community, and the time-bound community messing parameter δ.
Output: Temporal edges E.

$E \leftarrow \varnothing, E^e \leftarrow \varnothing$
$n'_v \leftarrow \sum_{i=1}^{n_c} |V_i|$
for $i = 1$ to n_c **do**
  Generate a tree with respect to V_i and store the edges in E_i
  for each u in V_i **do**
    $d_o \leftarrow$SampleForSource$(\pi_o, |V_i|, n_e \cdot \frac{|V_i|}{n'_v} - |V_i|)$
    for $e = 1$ to d_o **do**
      $v \leftarrow$SampleForTarget$(\pi_i, |V_i|, n_e \cdot \frac{|V_i|}{n'_v} - |V_i|, V_i)$
      $t \leftarrow$SampleForTimestamp$(\pi_t, w_i))$
      $E_i \leftarrow E_i \cup \{(u, v, t)\}$
    end for
    if Sample$(U(0,1)) \leq 1 - \delta$ **then**
      continue
    end if
    Get d_o^b using Equation (8)
    for $e = 1$ to d_o^b **do**
      $v \leftarrow$SampleForTarget$(\pi_i, n_v, n_e \cdot \frac{|V_i|}{n'_v} - |V_i|, \bigcup_{j=1}^{n_c} V_j)$
      $W \leftarrow \{w_i | \{u, v\} \cap V_i \neq \varnothing, i \in \mathbb{N}^+, i \leq n_c\}$
      $t \leftarrow$SampleForTimestamp$(\pi_t, \bigcup_{w \in W} w)$
      $E^e \leftarrow E^e \cup \{(u, v, t)\}$
    end for
  end for
  $E \leftarrow E \cup E_i \cup E^e$
end for
return E

As tree generation precedes the index construction, there may be conflicts between the remaining number of edges to be generated and the given out-degree distribution, i.e.,

$$n_e - n_v \neq n_v \alpha \sum_{k=(d_o)_{min}}^{(d_o)_{max}} k\pi_o(k) = \mathbb{E}[d_o], \tag{12}$$

where n_v denotes the number of nodes. We adapt $(d_o)_{max}$ to match $\mathbb{E}[d_o]$ with n_e.

The Equation 13 in the revised manuscript also has been updated as follow:

$$n'_e - n'_v = n'_v \alpha \sum_{k=(d_o)_{min}}^{(d_o)_{max}} k\pi_o(k). \tag{13}$$

B. R3-O2

Comment: The motivation behind the research problem is weak. It would greatly benefit from the inclusion of a real-life example to provide a tangible context and showcase the practical relevance of the problem being addressed.

Response: Thanks for your comment. The practical need for temporal datasets of associated algorithms motivates us to undertake the generation task featuring time-bound communities. We introduce the following **real-life example to provide a tangible context and showcase the practical relevance of the generation problem addressed in our paper**.

Example 1. *On community question answering (CQA) websites, the activeness of a discussion group may be confined to a specific time windows. Figure 5 (also presented in the manuscript as Figure 1b) depicts a real-life example network abstracted from an arbitrary page*[2] *of the Math Overflow site, where users are represented by nodes, and the time-stamped communications between them are represented by temporal edges. It can be clearly seen that the four individuals, Jins, Ian Agol, Derek Holt, and Johannes Hahn, discussed a specific mathematical issue only for a limited duration, from 5:18 AM to 11:04 AM on April 12th, 2022.*

As the example above illustrates, real communities may not only emerge but also step in to death, corresponding to specific time periods. The detection of such communities can be useful in many applications such as group-based recommendations [1] and network monitoring [2]. For example, if the four discussants Jins, Ian Agol, Derek Holt, and Johannes Hahn are detected to form a community on a specific topic in a time interval, information on similar topics can be timely multicasted to this community, instead of being separately sent to each individual. Additionally, other persons with comparable curiosities may be duly invited to join and thus broaden the discussion scope.

Other scenarios include money-laundering gang search on financial networks where criminals tend to transfer funds frequently within a short period and dynamic protein complex detection on protein interaction networks where proteins assemble and dissemble to respond to distinct biological functions in the cellular processes.

As discovering communities from temporal graphs has great values in many application scenarios, a series of associated algorithms emerged [2]–[15].

Despite the ongoing development in community discovery algorithms within temporal networks, the issue of needing ample data remains unresolved. We found that numerous algorithms [2]-[15] resorted to real but limited temporal datasets to validate their efficiency and effectiveness, where the numbers of nodes and edges do not exceed millions and tens of millions, respectively, in reported experiments. Moreover, we noticed that [6] and [7] utilized artificial temporal graph datasets for algorithm evaluation, but these are all limited in scope as well, with no dataset exceeding 1,000 nodes or 125 snapshots. In general, this data shortfall originates from two sources:

- *Difficulties in collecting real data.* The process of acquiring authentic data is fraught with multiple challenges. One primary issue is the inherent complexity of real temporal networks, which frequently exhibit extensive size both topologically and temporally. This complexity is further exacerbated in the case of major online social media platforms, encompassing billions of users and temporal interactions. Constructing an accurate real-time network from system logs requires substantial computational resources. Other significant barriers involves privacy concerns and corporate policies, which make it challenging to share and commonly use real data.
- *Disadvantages of existing temporal graph generation tech-*

[2]https://mathoverflow.net/questions/420198/algorithm-to-find-a-minimal-normal-subgroup-of-given-group-g-by-matrix-group-r

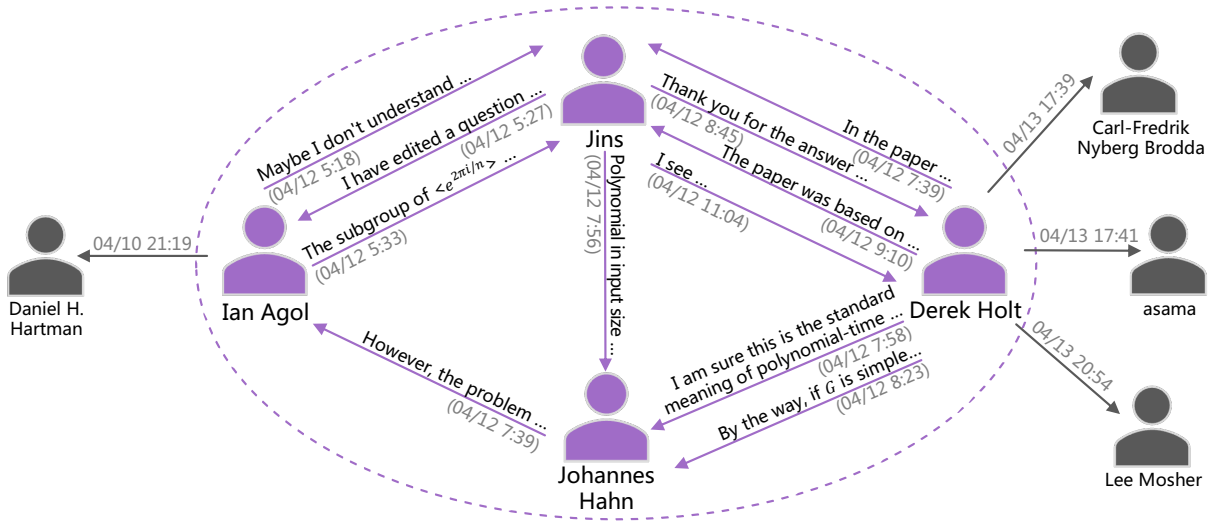


Fig. 5: A small Math Overflow discussion network in 2022.

niques. In terms of temporal graph generation, current methods either fail to accurately characterize the extinction of communities, or lack efficiency when tasked with generating largely scaled graphs that extend over considerable time spans.

Consequently, to address the demand for suitable extensive temporal datasets that facilitate comprehensive temporal networks analyses, it is imperative to develop an efficient and configurable method for the generation of temporal graphs that contain communities active within designated time intervals.

The GTB method we proposed can efficiently generate temporal graphs featuring time-bound communities, while preserving configurability for user-defined distributions. When discounting file input/output time, GTB can generate suitable datasets with ten billion temporal edges within one minute on a Ubuntu 16.04 server with a 20-core Intel Xeon E5-2630 CPU and 320 GB RAM. Larger-scaled data can also be generated by GTB within an acceptable time. This generation of extensive datasets presents an opportunity to assess the attributes of related algorithms, including efficacy and scalability.

To tangibly present our motivation, in Example 1 of the revised paper, we introduced the example discussion network apart from the toy financial network as follows:

Example 2. *Real communities in many scenarios correspond to specific time periods. In the context of online financial transactions, criminals tend to transfer funds frequently within a short period to evade the regulations of money laundering prevention [27]. Figure 1a provides a visual depiction of a money-laundering gang on the financial transfer network, where user accounts are represented as nodes, and the transactions between accounts at specific moments are represented by temporal edges. Specifically, the cluster of accounts labeled $\{A, B, C, D, E\}$ (marked as purple nodes) transferred money between each other multiple times (highlighted as purple links) from 15:31 to 15:46 on June 22nd, active for only 15 minutes. The scenario of online discussions about issues also steps on the similar pattern. To illustrate, Figure 1b (i.e., Figure 5 in this response) exhibits a real discussion group abstracted from a page[3] of the Math Overflow website. This discussion community formed by Jins, Ian Agol, Derek Holt, and Johannes Hahn, centered on a specific issue, started at 5:18 AM and ended at 11:04 AM on April 12th, 2022, only lasting for a limited duration.*

To clarify the motivation, in Section I of the revised paper, we also added the practical applications with respect to time-bound community detection and the corresponding data requirements:

The discovery of such communities can be useful in many applications such as community detection on community question answer sites like Math Overflow for group-based recommendation [28] and network monitoring [29], money-laundering gang search on financial networks [30]-[32], and protein dynamic protein complex detection on temporal protein interaction networks [33]-[36].
Despite the ongoing development in community discovery algorithms [29]-[40] within temporal networks, the practical need for synthetic datasets that reflect communities active within a time interval remains unresolved.

[1] P. Bedi and C. Sharma, "Community detection in social networks," *Wiley*

interdisciplinary reviews: Data mining and knowledge discovery, vol. 6, no. 3, pp. 115–135, 2016.
[2] M. Zhong, J. Yang, Y. Zhu, T. Qian, M. Liu, and J. X. Yu, "A unified and scalable algorithm framework of user-defined temporal $(k, \mathcal{X})$-core query," *IEEE Transactions on Knowledge and Data Engineering*, 2024.
[3] J. Yang, M. Zhong, Y. Zhu, T. Qian, M. Liu, and J. X. Yu, "Scalable time-range k-core query on temporal graphs," *Proceedings of the VLDB Endowment*, vol. 16, no. 5, pp. 1168–1180, 2023.
[4] C.-X. Zhu, L.-L. Lin, P.-P. Yuan, and H. Jin, "Discovering cohesive temporal subgraphs with temporal density aware exploration," *Journal of Computer Science and Technology*, vol. 37, no. 5, pp. 1068–1085, 2022.
[5] H. Qin, R.-H. Li, Y. Yuan, G. Wang, L. Qin, and Z. Zhang, "Mining bursting core in large temporal graphs," *Proceedings of the VLDB Endowment*, vol. 15, no. 13, pp. 3911–3923, 2022.
[6] D. Zhuang, J. M. Chang, and M. Li, "Dynamo: Dynamic community detection by incrementally maximizing modularity," *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 5, pp. 1934–1945, 2019.
[7] X. Ma and D. Dong, "Evolutionary nonnegative matrix factorization algorithms for community detection in dynamic networks," *IEEE transactions on knowledge and data engineering*, vol. 29, no. 5, pp. 1045–1058, 2017.
[8] R.-H. Li, J. Su, L. Qin, J. X. Yu, and Q. Dai, "Persistent community search in temporal networks," in *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE, 2018, pp. 797–808.
[9] Y. Li, J. Liu, H. Zhao, J. Sun, Y. Zhao, and G. Wang, "Efficient continual cohesive subgraph search in large temporal graphs," *World Wide Web*, vol. 24, pp. 1483–1509, 2021.
[10] Y. Tang, J. Li, N. A. H. Haldar, Z. Guan, J. Xu, and C. Liu, "Reliable community search in dynamic networks," *Proceedings of the VLDB Endowment*, vol. 15, no. 11, pp. 2826–2838, 2022.
[11] I. Moutidis and H. T. Williams, "Community evolution on stack overflow," *Plos one*, vol. 16, no. 6, p. e0253010, 2021.
[12] R. R. Rani, D. Ramyachitra, and A. Brindhadevi, "Detection of dynamic protein complexes through markov clustering based on elephant herd optimization approach," *Scientific reports*, vol. 9, no. 1, p. 11106, 2019.
[13] X. Shen, L. Yi, X. Jiang, Y. Zhao, X. Hu, T. He, and J. Yang, "Neighbor affinity based algorithm for discovering temporal protein complex from dynamic ppi network," *Methods*, vol. 110, pp. 90–96, 2016.
[14] M. Li, X. Meng, R. Zheng, F.-X. Wu, Y. Li, Y. Pan, and J. Wang, "Identification of protein complexes by using a spatial and temporal active protein interaction network," *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 17, no. 3, pp. 817–827, 2017.
[15] L. Ou-Yang, D.-Q. Dai, X.-L. Li, M. Wu, X.-F. Zhang, and P. Yang, "Detecting temporal protein complexes from dynamic protein-protein interaction networks," *BMC bioinformatics*, vol. 15, pp. 1–14, 2014.

C. R3-O3

Comment: The paper is challenging to follow due to the presence of numerous symbols without proper explanation when they are used. To enhance clarity and comprehension, it is necessary to provide thorough explanations and definitions for the symbols used throughout the paper.

Response: Thanks for your comment. We added an overall notation summary to provide thorough explanations for symbols used in the paper. To further improve the clarity and comprehension, we additionally updated our symbolic representations. The details are as follows:

1. Notation summary. We added an overall notation summary in tabular form **to provide proper explanations**, as shown in Table III.

2. Notation modification. **To enhance clarity and comprehension**, we simplified our symbolic notations. Specifically, we consolidated the symbols for the number of source nodes n_s and that of target nodes n_t into a single symbol for the number of nodes, denoted as n_v. Similarly for the member nodes of the i-th time-bound community, we denoted them collectively as V_i, instead of distinguishing between the source nodes V_s^i and target nodes V_t^i.

Moreover, we differentiated the parameters of the power-law distribution for community size and time window length by retaining λ for the former and introducing η for the latter.

TABLE III: Table of main notations.

Notation	Description
$G = (V, E)$	A temporal graph G
$w = [t_s, t_e]$	A time window w
n_v	The number of nodes
n_e	The number of temporal edges
n_c	The number of communities
π_o	The out-degree distribution
π_i	The in-degree distribution
π_t	The timestamp distribution
d_o^b	The between-out-degree of a node
λ	The power-law distribution parameter for community size
η	The power-law distribution parameter for the length of a time window
m	The number of overlapping relationships between time-bound communities
Ω	Time-bound community overlapping parameter
δ	Time-bound community messing parameter