Zhu et al. *EPJ Data Science* (2021) 10:30
https://doi.org/10.1140/epjds/s13688-021-00287-6

EPJ Data Science
a SpringerOpen Journal

REGULAR ARTICLE Open Access

Competition-driven modeling of temporal networks

Kaijie Zhu[1,2*], George Fletcher[1] and Nikolay Yakovets[2]

*Correspondence: k.zhu@tue.nl
[1]TU Eindhoven, Eindhoven, Netherlands
[2]NDSC, Zhengzhou, China

Abstract
We study the problem of modeling temporal networks constrained by the size of a concurrent set, a characteristic of temporal networks shown to be important in many application areas, e.g., in transportation, social, process, and other networks. We propose a competition-driven model for the generation of such constrained networks. Our method carries out turns of competitions along the timeline where each node in a network is labeled with a probability to gain outgoing edges in competitions. We present a thorough theoretical analysis to investigate the cardinality and degree distributions of the generated networks. Our experimental results demonstrate that our model simulates real-world networks well and generates networks efficiently and at scale.

Keywords: Temporal networks; Modeling; Concurrent set size

1 Introduction

Problem. Synthetic temporal networks are widely used to understand and verify the behavior of real networks [9, 22]. In recent years, various models have been proposed to generate synthetic networks satisfying certain characteristics. However, the distribution of the *concurrent set size* (CSS) [25] has not been considered in previous studies. CSS distribution refers to the mapping from a timestamp in a network domain to the number of edges *active* at this timestamp.[1] In many real-world networks, the CSS follows one or more distributions. Consider an example shown in Fig. 1. The CSS of transport trips by free hired vehicles [3] in New York City follows a Poisson distribution in the morning and then a normal distribution for the rest of the day. Another example is the CSS of the raw dataset used in BPI challenge 2014 [2], which follows a linear-transformed power-law distribution in most of the time domain. These examples show that studying the behavior of the CSS can provide a better understanding of real-world networks. Further, modeling of the CSS can aid in the generation of realistic synthetic temporal networks. Finally, a deeper understanding of CSS can provide guidance towards efficient network processing approaches [24].

[1]An edge e is active at a certain timestamp t when t falls in between e's starting and ending times.

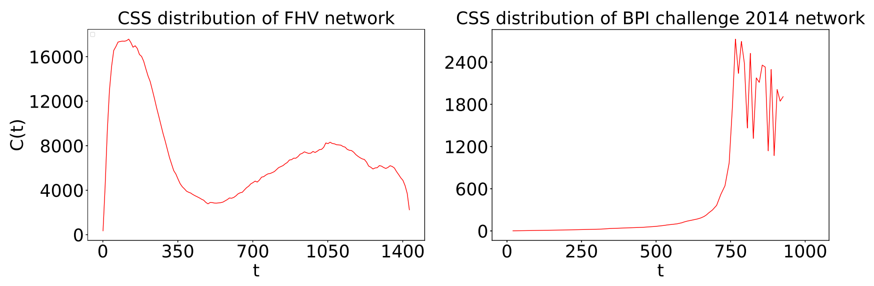

Figure 1 The CSS distribution of some real networks. $C(t)$ is the CSS at time t. The FHV CSS (left) can be estimated as a Poisson distribution when $t \leq 480$ and a normal distribution in the rest. The BPI CSS (right) can be estimated as a linear-transformed power-law distribution when $t \leq 800$

Contributions. In this paper, we directly address the generation issues in the CSS-constrained temporal networks, particularly:

- We propose a new model, namely a *competition-driven* model (CDM), to generate the CSS-constrained networks.
- We present a theoretical analysis of the CDM and show how CDM affects several important characteristics of generated networks.
- We carry out an in-depth experimental study which demonstrates that CDM can simulate the real-world networks well and that its generation process is scalable.

Organization. The rest of this paper is organized as follows. Section 2 presents related work. Section 3 presents our proposed model. Section 4 gives a theoretical analysis of our model and demonstrates its characteristics. Section 5 presents experiments that evaluate the performance of our model. Section 6 concludes the paper with a summary of our findings.

2 Related work

Activity-driven network (ADN) model, proposed by Perra et al. [17], is the most well-studied network model in the state of the art. This model initializes each node v with a firing rate a_v drawn from a given probability distribution $F(x)$. At each timestamp t and with probability a_v, node v becomes active and generates m instant outgoing edges linked to the other nodes randomly. Several studies have been carried out to extend the model in both structural and temporal fields. For structural extensions, prior studies concentrate on the selection of the edge destination [6, 13, 14, 21]. Alessandretti et al. [6] extend each node with an *attractiveness* value representing its probability to be selected as the destination of edges. Other works extend the model with a reinforcement mechanism, which exhibits the preference of nodes to connect to previously contacted nodes [13, 14, 21]. Another collection of works concentrate on the incorporation of community structure [14, 16]. Laurent et al. [14] introduce focal closure and cyclic closure, which gives rise to the community structure in the network. Nadin et al. [16] initialize each node to a community. In each turn, a node could either connect other nodes within (or outside) the same community with probability μ (or $1 - \mu$). For temporal extension, Sunny et al. [20] introduces the duration for edges so that edges are lasting entities rather than instant ones.

Besides ADN, there are also other categories of temporal network generation models. The Renewal process model extends the Gillespie algorithm [11] to model the network generation where each node is modeled as a Poisson process and the superposed nodes are regarded as the inter-event time distribution [8, 15]. Holme [12] and Speidel et al. [18] model the generation as labeling of static links with temporal aspects. Starnini et al. [19] and Zhang et al. [23] model the generation as a process involving agents performing a random-walk in the unit square. Each agent interacts with its neighbors every time a random walk is performed. To deal with situations where information of entities is missing, Cho et al. [10] proposes a self-exciting process model where the event rate between each pair of entities is modeled as a Hawkes process.

To the best of our knowledge, there is no research on modeling and generation of CSS-constrained networks. In the next section, we give the preliminaries and the details of our model.

3 Model

We next present the basic definitions and state the network generation problem studied in this paper. Then, we propose our competition-driven model which serves as a solution to the stated problem.

3.1 Preliminaries

Temporal network. A temporal network is modeled as a graph $G = (V, E, T)$, where V and E are respectively the set of nodes and temporal edges in G. T is a temporal domain of the network. Each edge $e \in E$ is represented by a tuple (u, v, t_s, t_e) where (1) $u, v \in V$ represent a directed link from node u to v, denoted (u, v); (2) t_s, t_e is a pair of timestamps s.t. $t_s \leq t_e$, representing the active lifespan of link (u, v), denoted $[t_s, t_e]$.

Activity behavior. We start by presenting the definition of nodes' activity behavior. Given a node $v \in V$ and time $t \in [1, T]$, we say v is active at t if there exists $e \in E$ such that $e.u = v$ and $e.t_s = t$. Otherwise, we say v is inactive at t. Also, we define edges' activity behavior. Given an edge $e \in E$ and time $t \in [1, T]$, we say e is active at t if $t \in [e.t_s, e.t_e]$ (or is an active edge at t) and ends at time $e.t_e + 1$. Note that multiple active edges are allowed between the same pair of nodes at an arbitrary time t.

Snapshot. In order to obtain an instant status of a temporal graph, $G = (V, E, T)$ could also be viewed as a sequence of static graphs $G = \{G(1), G(2), \ldots, G(T)\}$, where $G(1), G(2), \ldots, G(T)$ respectively represents the instant status of G at $t = 1, 2, \ldots, T$. We call $G(t)$ a snapshot at time t. Formally, given a temporal network $G = (V, E, T)$ and a timestamp $t \in [1, T]$, the corresponding snapshot is represented as $G(t) = (V, E(t))$, where $E(t) = \{e_1^t, e_2^t, \ldots, e_n^t\} \subseteq E$ is the set of active edges at t.

CSS distribution. Given a temporal graph $G = \{G(1), G(2), \ldots, G(T)\}$, a CSS distribution is represented as a function $C(t) = |E(t)|, \forall t \in [1, T]$. That is, the CSS distribution indicates the number of active edges at each timestamp t.

Problem statement. We study the problem of how to generate CSS-constrained networks. Given a set of nodes V and the target CSS distribution $C(t)$, we aim to generate a temporal network $G = \{G(1), G(2), \ldots, G(T)\}$ with node set V, where $|E(t)| = C(t)$ for $\forall t \in [1, T]$.

Table 1 Overview of the important characteristics of the generated networks

Number of generated edges	$\|E\|$
Relative degree	$A(v)$
Inter-event time distribution	$\mathcal{I}$
Duration distribution	$\mathcal{D}$
CSS distribution	$C(t)$

[b]

Table 2 Overview of the parameters used in the proposed model

Nodes set	V
Power value distribution	f
Inter-event time distribution	$\mathcal{I}$
Duration distribution	$\mathcal{D}$
CSS distribution	$C(t)$

Besides the values for $C(t)$ and V, we consider additional characteristics of generated networks as the structural and temporal characteristics of real networks are heterogeneous. For example, inter-event time (IET) in some real networks follows a power-law distribution with notable heavy tail [15, 18, 21]. This makes the IET distribution an important and necessary parameter in network generation. Similar results could also be found for the real degree of nodes [16]. We summarize the important characteristics of our generated networks in Table 1.

Relative degree. For convenience of comparison and analysis, degree of a node needs to be stable in its distribution across networks of different sizes. For this purpose, we define a *relative* degree of a node as follows. Given a temporal graph $G = (V, E, T)$ and $\forall v \in V$, we call $A(v) = \frac{\delta(v)}{|E|}$ the *relative* degree of v, where $\delta(v)$ is the number of edges outgoing from v. As defined, $A(v)$ denotes the proportion of edges starting from a given node v.

Inter-event time (IET) distribution. The distribution captures the activity behavior of nodes. Given temporal graph $G = (V, E, T)$ and $\forall v \in V$, we collect the distinct start times of edges outgoing from v, denoted $\theta(v) = \{t_1^v, t_2^v, \ldots, t_\epsilon^v\}$ where $t_i^v \in [1, T]$ and $t_i^v < t_{i+1}^v$. For $i \in [1, \epsilon)$, we call $\tau_i(v) = t_{i+1}^v - t_i^v$ an inter-event time (IET). We assume that τ follows a probability distribution $\mathcal{I}(f, \underline{\tau}, \overline{\tau})$, where f is a parameter distribution function, $\underline{\tau}$ and $\overline{\tau}$ are minimum and maximal IETs respectively.

Duration distribution. The distribution captures the activity behavior of edges. Given temporal graph $G = (V, E, T)$ and $\forall e \in E$, we call $d(e) = e.t_e - e.t_s + 1$ the *duration* of edge e. We assume that edge duration d follows a distribution $\mathcal{D}(f, \underline{d}, \overline{d})$, where f is a parameter distribution function, $\underline{d}$ and $\overline{d}$ are minimum and maximum edge durations, respectively.

3.2 Competition-driven model

In this work, we propose the competition-driven model (CDM) to generate the CSS-constrained networks accurately and efficiently. Table 2 presents the input parameters used in our model. In CDM, each node is associated with a power value $\Pi(v)$ and next active time $nat(v)$. The former determines v's strength in edge generation while the latter determines its next time to be active. That is, a node with higher $\Pi(v)$ has a higher chance to become the source of newly generated edges at time $nat(v)$. Values for $\Pi(v)$ are drawn from a parameter probability distribution f (e.g., the power value distribution) and values for $nat(v)$ are drawn from the IET distribution $\mathcal{I}$.

Network generation. Using the above model, the procedure of network generation is shown in Algorithm 1. A dedicated *active-list* structure named *Active* is maintained to store the set of active edges in real-time. The basic idea of network generation is traversing the CSS distribution $C(t)$ in time and adjusting the size of *Active* (denoted $|Active|$) according to $C(t)$ at a certain timestamp t. For this aim, we define the following two basic operations to maintain *Active*.

- $InsActive(Active, e)$: insert the edge e into *Active*; Return 1 for success and 0 for failure.
- $DelActive(Active, t)$: delete all edges e s.t. $t > e.t_e$ from *Active*; Return the collection of deleted edges.

Algorithm 1: The network generation using CDM

```
Input: Nodes set $V$, power value distribution $f$, IET distribution $\mathcal{I}$, duration
       distribution $\mathcal{D}$, CSS distribution $C(t)$
Output: Synthetic temporal graph $G = (V, E, T)$
1  Initialize $\Pi(v)$ and $nat(v)$ for each $v \in V$ by using $f$ and $\mathcal{I}$
2  $t \leftarrow \min_{C(t_i)\neq\emptyset} t_i$, $T \leftarrow \max_{C(t_i)\neq\emptyset} t_i$
3  while $t \leq T$ do
4      $D \leftarrow DelActive(Active, t)$
5      $E \leftarrow E \cup D$
6      $n \leftarrow C(t) - |Active|$
7      if $n < 0$ then
8          $D \leftarrow PruneActive(Active, t-1, -n)$
9          $E \leftarrow E \cup D$
10     else
11         Collect the participants set $\Gamma(t) = \{p_1^t, p_2^t, \dots, p_k^t\}$
12         $i \leftarrow 1$
13         while $i \leq k$ do
14             $\mathcal{S}_t(p_i^t) \leftarrow \Pi(p_i^t) / \sum_{n=1}^{k} \Pi(p_n^t)$
15         while $n > 0$ do
16             Draw a participant $p$ from $\mathcal{S}_t$ as source.
17             if it is the first time for $p$ to be drawn in this turn then
18                 Draw an IET $\tau$ from $\mathcal{I}$
19                 $nat(p) \leftarrow t + \tau$
20             Draw a duration $d$ from $\mathcal{D}$.
21             Draw a destination $v$ from $V - \{p\}$
22             $InsActive(Active, (p, v, t, t + d - 1))$
23             $n \leftarrow n - 1$
24     $t \leftarrow t + 1$
25 $E \leftarrow E \cup Active$
26 return $(V, E, T)$
```

Algorithm 2: PruneActive

```
Input: active list $Active$, timestamp $t$, number of pruned edges $n$
Output: the set of pruned edges $D$
1 $D \leftarrow set$
2 while $n > 0$ do
3     $e \leftarrow$ the 1st edge in $Active$
4     $D \leftarrow D \cup \{(e.u, e.v, e.t_s, t)\}$
5     $Active \leftarrow Active - \{e\}$
6     $n \leftarrow n - 1$
7 return $D$
```

For ease of maintenance, edges in *Active* are sorted by their end-time in ascending order. In this way, the complexity of *InsActive* and *DelActive* is logarithmic in $|Active|$. With the structure, the specific operations to be carried out at any given time t could be determined: when $C(t)$ is smaller than $|Active|$, some of the existing edges should be forcibly deactivated and removed from *Active* in order to satisfy $|Active| = C(t)$ constraint. We define one additional operation for *Active* in order to deal with this situation:

- *PruneActive*($Active, t, k$): select k edges, set their end-time to t, and delete them from *Active* Return the collection of deleted edges.

The procedure of *PruneActive* in our work is shown in Algorithm 2. Here, we apply the end-time-first pruning strategy to prune *Active*. That is, we select the top-k edges with minimal end-time from *Active*, reduce their end-time to t, and delete them from *Active*. Various pruning strategies can be used in *PruneActive*. We choose end-time-first-pruning for the following reasons. First, this strategy provides the best efficiency because edges in *Active* are sorted by their end-time. Second, end-time-first-pruning also helps to preserve the duration distribution in the generated network, which is a desirable network characteristic.

Additionally, a total of $n = C(t) - |Active|$ edges should be generated and inserted into *Active*. The algorithm first collects the set of nodes $\Gamma(t) = \{p_1^t, p_2^t, \dots, p_k^t\}$ with $nat(v) \leq t$. We call these nodes in the collection *participants* at current time t.[2] Continuously, the algorithm constructs a probability distribution $\mathcal{S}_t(p)$ by normalizing $\Pi(p_i^t)$ for each $i \in [1, k]$. We call $\mathcal{S}_t(p)$ the competition distribution and it reveals the probability for each participant to "win" in each turn of the coming competition at time t. With the constructed $\mathcal{S}_t(p)$, the algorithm carries out a n-turn competition to generate new edges. In each turn, a participant $p \in \Gamma(t)$ is first selected as the source of link according to $\mathcal{S}_t(p)$. Next, a duration d is generated from duration distribution $\mathcal{D}$, and another node v is selected uniformly from the remaining nodes as the destination In this way, a new temporal edge $(p, v, t, t + d - 1)$ is created and inserted into *Active*. And if it is the first time for p to win in this turn, algorithm updates $nat(p)$ to $t + \tau$, where τ is drawn from $\mathcal{I}$ to determine its next time to be active. Similar turns are repeated until n turns have been carried out, which means n new edges have all been created in this competition. Note that if p does not win any turns in the competition, $nat(p)$ is not updated and p would be continuously considered as a

[2]If there is no $v \in V$ s.t. $nat(v) \leq t$, we collect the set of nodes u such that $nat(u) - t \leq \omega \cdot (t - nat'(u))$ and set each $nat(u)$ to t, where threshold $\omega \in (0, 1.0]$ and $nat'(u)$ is the last active time of u

participant in the competition at the next timestamp. This way, p's IET is prolonged until it can win at least one turn in a competition.

The iterative competitions are repeatedly carried out until $C(t)$ is completely traversed in time. The complexity of the generation algorithm is $O(T \cdot |V| + |E| \cdot \log |E|)$, where $|E|$ is the total number of generated edges. Note that $|E|$ is not an input parameter to the CDM and its exact value can only be known when the network is completely generated. Similarly, for each node $v \in V$, relative degree $A(v)$ is also known after the generation since they depend on $|E|$. In the following section, we present the theoretical analysis of how values for $|E|$ and $A(v)$ in the produced networks are influenced by the generation algorithm.

4 Analysis

Two natural questions about the CDM are: (1) As the number of edges $|E|$ is not an input parameter to the algorithm, what is the expected cardinality for the generated network? (2) Similarly, what would the relative degree $A(v)$ be like? Answers to these questions respectively help to evaluate the necessary storage cost for generation and investigate the structural characteristics of the generated network. In this section, we provide an analysis to answer these two questions. For ease of analysis, we make the assumption that the activity behavior of both $v \in V$ and $e \in E$ follow the Poisson process and $\mathcal{I}$, $\mathcal{D}$ follow exponential distributions with λ_1, λ_2 parameters, respectively. Besides, we assume each participant in a competition can win at least one turn so that their IETs are not prolonged and follow $\mathcal{I}$ strictly.

Cardinality. For $t \in [1, T]$, let $O(t)$ demonstrate the number of edges that should be generated at timestamp t. The equation to describe the relation between network cardinality $|E|$ and $O(t)$ could be written down as follows:

$$|E| = \sum_{t=1}^{T} O(t). \tag{1}$$

Let $R(t)$ demonstrate the number of remaining edges at time t after *DelActive* is invoked. The equation to describe $O(t)$ is as follows:

$$O(t) = \begin{cases} C(t) - R(t) & C(t) > R(t), \\ 0 & C(t) \leq R(t). \end{cases} \tag{2}$$

That is, given timestamp t, $O(t)$ merely contributes to the cardinality when $C(t) > R(t)$. For example, Fig. 2 presents a collection of edges generated using the CDM and $C(t) = \{0:1, \dots, 3:1, \dots, 5:3, 6:3, 7:4, 8:3\}$. Each edge is represented by its interval. Consider the edge generation at $t = 7$ is completed and we are going to generate the collection of edges at $t = 8$. *Active* at $t = 7$ contains the edges e_1, e_2, e_3, e_4. Then *DelActive* deletes e_1, e_2 from *Active* since they both end at $t = 8$. In this way, only two edges e_3, e_4 survive in *Active* after the edge deletion, hence $R(8) = 2$. Since $C(8) = 3$ and Equation (2) gives $O(8) = C(8) - R(8) = 1$, this means that a single edge needs to be generated at $t = 8$.

As cardinality analysis is generally used for network storage and construction time estimation, here we use the worst-case method to estimate the output of Equation (1). In this

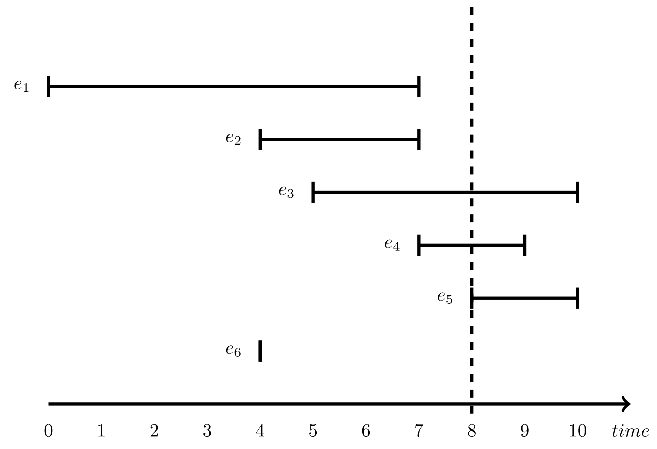

Figure 2 An example of an edge generation by the CDM. Dashed line corresponds to the $C(8) = 3$

worst case, we assume that $R(t)$ is always smaller than $C(t)$. Then, the worst-case equation for the maximum number of generated edges $|E|_m$ is described as follows:

$$|E|_m = \sum_{t=1}^{T} \bigl(C(t) - R(t)\bigr). \tag{3}$$

From Algorithm 1, we know that $R(t)$ consists of the set of edges active at both $t-1$ and t. Then, $R(t)$ can be computed as follows:

$$R(t) = C(t-1) \cdot \bigl(1 - P_d(t)\bigr), \tag{4}$$

where $P_d(t)$[3] represents the probability for each $e \in E(t-1)$ to end at time t. In our example, we can estimate that $P_d(8)$ is approximately 0.5 since there is $C(7) = 4$ and $R(8) = 2$. Here, we use the knowledge of the stochastic process to make further deduction on $P_d(t)$. Considering a probability event ϵ, Poisson process uses the following equation to express and compute the probability that ϵ *happens k times in duration* $[t, t+\tau]$:

$$P\bigl[N(t+\tau) - N(t) = k\bigr] = \frac{e^{-\lambda\tau}(\lambda\tau)^k}{k!}. \tag{5}$$

Note that $P_d(t)$ can be also described as the probability that *an edge active at* $t-1$ *is going to end at time t*. Based on our assumed Poisson process for e and exponential distribution for $\mathcal{D}$, $P_d(t)$ can be transformed into the following:

$$P_d(t) = P\bigl[N(t) - N(t-1) = 1\bigr] = \lambda_2 e^{-\lambda_2}. \tag{6}$$

By substituting Equation (6), (4) into (3), we could obtain the following equation of describe the expected cardinality for synthetic network.

$$|E|_m = C(T) + \sum_{t=1}^{T-1} C(t) \cdot \lambda_2 e^{-\lambda_2}. \tag{7}$$

[3]Since edges share the same $\mathcal{D}$ in the CDM, the ending probability is the same for $e \in E$.

With this equation, the complexity of CDM becomes more intuitive. Also, maximal memory cost in network generation can be evaluated.

Relative degree. Next, we give the derivation of the relative degree $A(v)$. The equation to describe $A(v)$ is as follows:

$$A(v) = \frac{\sum_{t=1}^{T} o(v,t)}{|E|}, \tag{8}$$

where $o(v,t)$ is the number of generated outgoing edges starting from v at time t. The value of $o(v,t)$ relies on whether v is active at t. Based on our assumption and letting $P_a(t)$ [4] be the probability for $v \in V$ to be active at t, the equation is as follows:

$$P_a(t) = P\big[N(t) - N(t-1) = 1\big] = \lambda_1 e^{-\lambda_1}. \tag{9}$$

In this way, the equation to describe $o(v,t)$ is as follows:

$$o(v,t) = \begin{cases} 0 & \text{with probability } p = 1 - \lambda_1 e^{-\lambda_1}, \\ \mathcal{S}_t(v) \cdot O(t) & \text{with probability } p = \lambda_1 e^{-\lambda_1}. \end{cases} \tag{10}$$

According to Algorithm 1, $\mathcal{S}_t(v)$ could be computed as follows:

$$\mathcal{S}_t(v) = \left(\Pi(v) \Big/ \sum_{i=1}^{|\Gamma(t)|} \Pi\big(p_i^t\big) \right). \tag{11}$$

By substituting Equation (11) into (10), we could obtain:

$$o(v,t) = \begin{cases} 0 & \text{with } p = 1 - \lambda_1 e^{-\lambda_1}, \\ \frac{\Pi(v) \cdot O(t)}{\sum_{i=1}^{|\Gamma(t)|} \Pi(p_i^t)} & \text{with } p = \lambda_1 e^{-\lambda_1}. \end{cases} \tag{12}$$

The combination of Equations (8) and (12) reveals that in order to analyze $A(v)$, we only need to concentrate on the $o(v,t)$ in which v is active at timestamp t. We use $B(v) = \{b(v,1), \ldots, b(v,k), \ldots\}$ to demonstrate the collection of v's active timestamps $b(v,k)$ represent the kth active time of v. Equation (8) could be simplified into following format:

$$A(v) = \frac{\sum_{k=1}^{|B(v)|} o(v, b(v,k))}{|E|}. \tag{13}$$

Aligning Equations (13) with (12), we could obtain the following equation which illustrates the factors impacting $A(v)$:

$$A(v) = \frac{1}{|E|} \cdot \sum_{k=1}^{|B(v)|} \frac{\Pi(v) \cdot O(b(v,k))}{\sum_{i=1}^{|\Gamma(b(v,k))|} \Pi(p_i^{b(v,k)})}. \tag{14}$$

Equation (14) reveals that the relative degree of node v is influenced by following factors:

[4]Since nodes share the same $\mathcal{I}$ in the CDM, the active probability is the same for all $v \in V$.

- $|B(v)|$, the number of timestamps when v is active (i.e., the number of competitions v participated). The larger $|B(v)|$ provides more opportunities for v to earn outgoing edges.
- $|\Gamma(t)|$, the number of participants in competition at time t. The larger $|\Gamma(t)|$ tends to weaken $\mathcal{S}_t(v)$, which in turn leads to less outgoing edges from v.
- $\Pi(v)$, the power value of v. The larger $\Pi(v)$ tends to enhance $\mathcal{S}_t(v)$, which in turn leads to more outgoing edges from v.
- $O(t)$, the number of generated edges at time t. The larger $O(t)$ leads to more outgoing edges from v when $\mathcal{S}_t(v)$ is fixed.

In order to mine more underlying factors on $A(v)$, we introduce the mean-field method to simplify the variables in the model and regard the inferred result as the benchmark. Let $\overline{A(v)}$ be the mean static degree of node v. The equation to describe the mean field is as follows:

$$\overline{A(v)} = \frac{1}{|E|} \cdot \sum_{k=1}^{\overline{B}} \frac{\Pi(v) \cdot \overline{O}}{\sum_{i=1}^{\overline{\Gamma}} \Pi(p_i^{b(v,k)})} = \frac{\overline{O}}{|E|} \cdot \sum_{k=1}^{\overline{B}} \frac{\Pi(v)}{\sum_{i=1}^{\overline{\Gamma}} \Pi(p_i^{b(v,k)})}, \tag{15}$$

where $\overline{B}$ is the mean number of competitions v participated. $\overline{\Gamma}$ is the mean number of participant at time t. $\overline{O}$ is the mean number of edges that should be generated at time t. Corresponding equations to describe these mean-field parameters are as follows:

$$\overline{B} = E\big[B(v)\big] = \lambda_1 e^{-\lambda_1} T, \tag{16}$$

$$\overline{\Gamma} = E\big[\big|\Gamma(t)\big|\big] = \lambda_1 e^{-\lambda_1} |V|, \tag{17}$$

$$\overline{O} = |E|/T. \tag{18}$$

By substituting equation (16), (17), (18) into (15). The mean degree equation could be transformed as follows:

$$\overline{A(v)} = \frac{1}{T} \cdot \sum_{k=1}^{\lambda_1 e^{-\lambda_1} \cdot T} \frac{\Pi(v)}{\sum_{i=1}^{\lambda_1 e^{-\lambda_1} \cdot |V|} \Pi(p_i^{b(v,k)})}. \tag{19}$$

Equation (19) reveals the two characteristics of the relative degree in our model: first, as the number of nodes $|V|$ increases, the relative degree of each node will drop because $|V|$ determines the sum of cumulative adding in denominator. Second, given the number of nodes $|V|$, more involved participants make the distribution of $A(v)$ much closer to $\Pi(v)$ as it makes $\sum_{i=1}^{|\Gamma(t)|} \Pi(p_i^{b(v,k)})$ closer to 1. That is, relative degree $A(v)$ is exactly reflected by $\Pi(v)$ in the most ideal situation. The larger nodes set size makes $A(v)$ closer to $\Pi(v)$.

5 Experimental evaluation

In this section, we present our experimental investigation for the CDM. We aim to answer the following questions. First, we would like to know if CDM could simulate real networks with both structural and temporal characteristics preserved. Second, we investigate to what extent various graph configuration parameters influence the synthetic networks generated by the CDM.

[b]

Table 3 Overview of the real-world networks used in the experiments. In each network, nodes represent the locations and edges represent the vehicle trips between pairs of locations. Each edge is associated with an interval for its period of activity

Name	$\lvert V\rvert$	$\lvert E\rvert$	T
Yellow	253	236,522	1440 minutes
FHV	261	585,691	1440 minutes
HVFHV	260	823,629	1440 minutes
Flight	335	566,942	744 hours

5.1 Setup

Environment Our experiments are carried out on a server with 192 GB RAM and 2 Intel(R) Xeon(R) CPU X5670 with 6 cores at 2.93 GHz running a Linux operating system. We implemented the in-memory versions of the CDM in C++.

Datasets We consider four real networks in the transportation domain: Yellow, FHV, HVFHV, and Flight. In each network, nodes represent the locations and edges represent the vehicle trips between pairs of locations. Each trip is associated with an interval for its period of activity. Yellow [1] records the trips on the yellow taxi in New York City on 2 January 2018 and each trip is labeled with an interval to represent its duration. FHV [3] records the trips on for hired vehicles in New York City on 1 January 2018. HVFHV [5] records the trips on high-volume for hired vehicles on 1 June 2019. Flight [4] records the trips on airlines in the US in January 2019. An overview of statistics of the four networks is given in Table 3 and their CSS distributions are shown in Fig. 3.

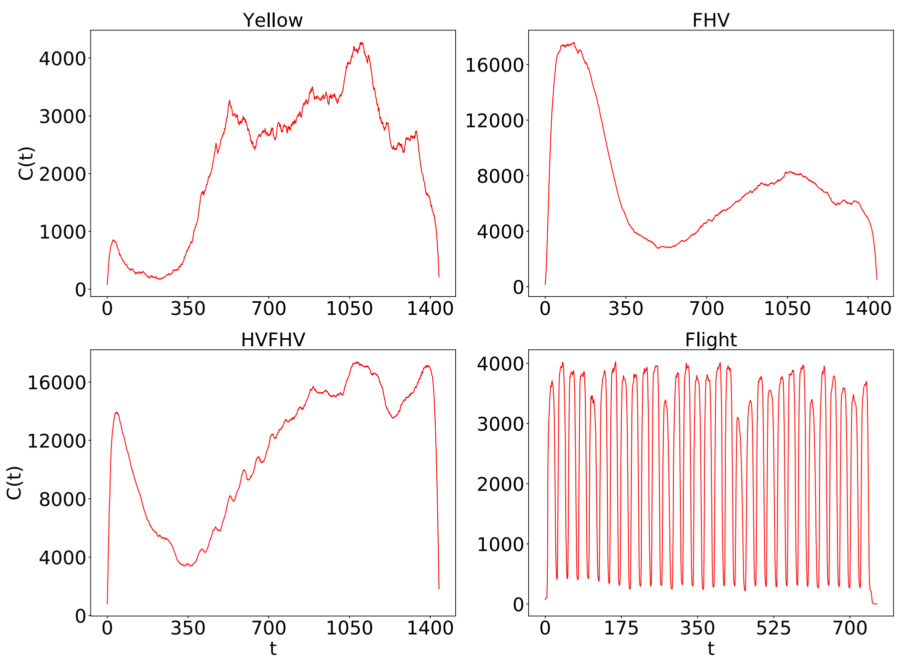

Figure 3 CSS distribution of the real-world networks used in the experiments

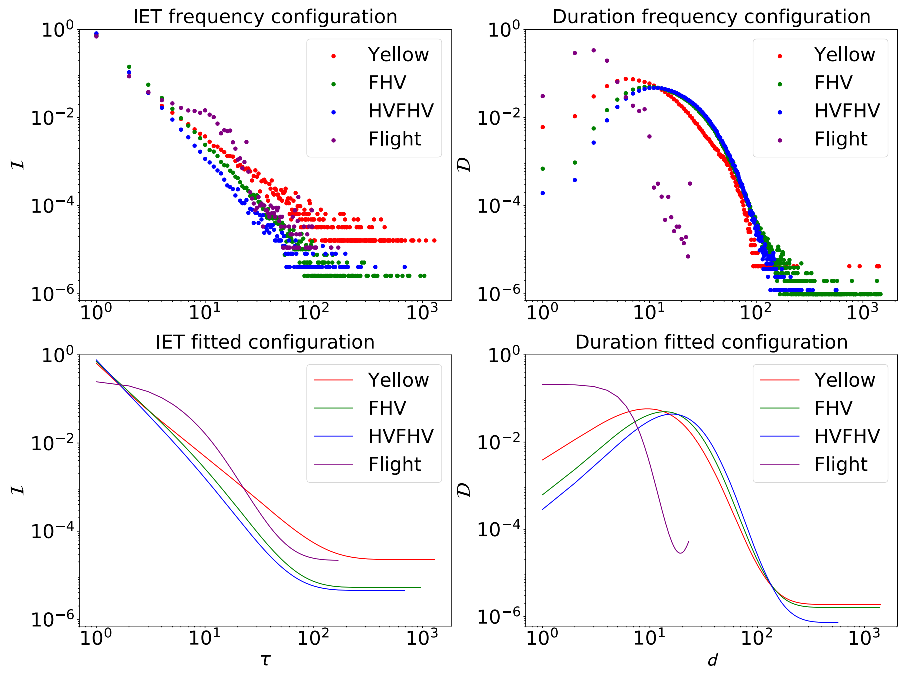

Figure 4 The IET and duration values used in frequency and fitted configuration. In frequency configuration (upper left and right), statistical estimations of real measures are directly used as input schema. In fitted configuration, IET (lower left), duration (lower right) distributions are used directly as estimated from real networks, and the fitted result are used for $\mathcal{I}$ and $\mathcal{D}$ distributions

Experiments We run two categories of experiments to investigate the performance of CDM. The first category of experiments deals with the *quality* of network simulation by the CDM. We investigate real networks' relative degree, IET, duration, and CSS distribution to obtain a graph configuration to be used in network generation. Specifically, we obtain graph configurations in two ways. The first method is called the *frequency* configuration, in which statistical estimations of real measures are directly used as input schema. The second method is called the *fitted* configuration, in which for IET and duration distributions are used directly as estimated from real networks and we use the fitted result for $\mathcal{I}$ and $\mathcal{D}$ distributions. We use the power-law cut-off model $y = k \cdot \tau^{\alpha} \cdot e^{-\frac{\tau}{\tau_c}} + h$. The values used in frequency and fitted configuration for each network are shown in Fig. 4. The parameter k is the CSS coefficient which represents the times that basic CSS value is enlarged.

The second category of experiments investigates the *scalability* of the CDM. We use instances with two types of CSS: (1) the linear $C(t) \sim t$ and (2) the Gaussian $C(t) \sim N(702, 180.0^2)$. The former case aims to investigate the CDM performance in monotonic increasing CSS and the later case aims to investigate the CDM performance in non-monotonic CSS. The default setup for the remainder of the configuration is shown in Table 4. These configuration parameters are either popularly used in existing benchmark for network modeling and generation [7] or supposed to impact the underlying structures in networks. To investigate such impact in various networks generated by using CDM, we vary the configuration parameters as follows. (1) We set $|V|$ in $[500, 750, 1000, 1250, 1500]$ to investigate the nodes cardinality impact in CDM. By setting various $|V|$, we obtain the networks involving either more or less entities. (2) We set CSS coefficient k

Table 4 The default configuration for the scalability experiments

Schema	Value	Description
$\lvert V\rvert$	500	number of nodes
$\lvert E\rvert$	200,000,000	number of edges
f	$\sim x^{-1.5}$	power value distribution
$\mathcal{I}$	$\sim \tau^{-1.5}, \underline{\tau} = 1, \overline{\tau} = 1000$	IET distribution
$\mathcal{D}$	$\sim d^{-1.5}, \underline{d} = 1, \overline{d} = 1000$	duration distribution
k	50×10^7	coefficient in $C(t)$

in $[1, 5, 50, 500, 5000] \times 10^7$ to investigate the CSS coefficient impact. By setting various k, we obtain the networks with either higher or lower CSS value at each timestamp. (3) We set $|E|$ in $[20, 40, 60, 80, 100]$ million to investigate the edges cardinality impact.[5] By setting various $|E|$, we generate either small or large networks. (4) We set $\overline{\tau}$ in $[1, 10, 100, 1000, 10{,}000]$ to investigate the IET impact. By setting various $\overline{\tau}$, the intensity of IET heavy tail (i.e., the existence of long IETs) in generated graph can be controlled. (5) We set $\overline{d}$ in $[1, 10, 100, 1000, 10{,}000]$ to investigate the duration impact. By setting various $\overline{d}$, the existence of lasting edges can be controlled.

We use six measures to evaluate the result: the distribution of (1) network generation time, (2) relative degree, (3) closeness, (4) IET, (5) duration, and (6) stability [22]. Given a node $v \in V$, the *closeness* is a measure of how close v is to any other nodes in the network. The measure is computed as the inverse of the average distance from v to any other nodes in the network, which is shown as follows:

$$closeness(v) = \frac{|V| - 1}{\sum_{u \in V - \{v\}} dist(v, u)}, \tag{20}$$

where $dist(v, u)$ represents the minimal distance (i.e., number of hops) from node v to u. The *stability* is a summary of v's evolving degree structure in time, which is measured based on the notion of *degree rank*. Given the set of snapshots $\{G(1) \ldots G(T)\}$, the *degree rank* of v in snapshot $G(t)$ is computed as follows.

$$rank(t, v) = \frac{\delta_t(v)}{\max_{u \in V} \delta_t(u)}, \tag{21}$$

where $\delta_t(v)$ is the number of out-going edges from v in $G(t)$. With these notions, given the set of snapshots $\{G(1) \ldots G(T)\}$ the *stability* of v is computed as follows.

$$stability\bigl(v, \bigl\{G(1) \ldots G(T)\bigr\}\bigr) = 1 - 2\sigma_v, \tag{22}$$

where σ_v is the standard deviation of v's degree rank over all snapshots. As a trade-off between measuring accuracy and efficiency, for each network in this experiment, we compute the *stability* based on 1000 uniformly selected snapshots.

5.2 Results and analysis

Quality of network simulation. Table 5 reports the generation time for the two types of simulations (frequency and fitted configurations) for different networks. We note that

[5] Every time $C(t)$ is consumed, we return to $C(0)$ and continue the generation iteration, until the desired cardinality is reached.

Table 5 Generation time of simulation result. The number of edges in generated networks are close to the cardinality in original real networks. The largest network with 800K edges (HVFHV-frequency) can be generated in no more than 15 seconds

Name	$\|E\|$	construction cost (ms)
FHV-frequency	563,243	12,176.3
FHV-fitting	533,572	12,574.7
Yellow-frequency	237,275	7859.7
Yellow-fitting	199,298	8770.9
HVFHV-frequency	826,523	14,980.0
HVFHV-fitting	755,707	14,714.8
Flight-frequency	569,483	10,717.4
Flight-fitting	583,964	11,266.1

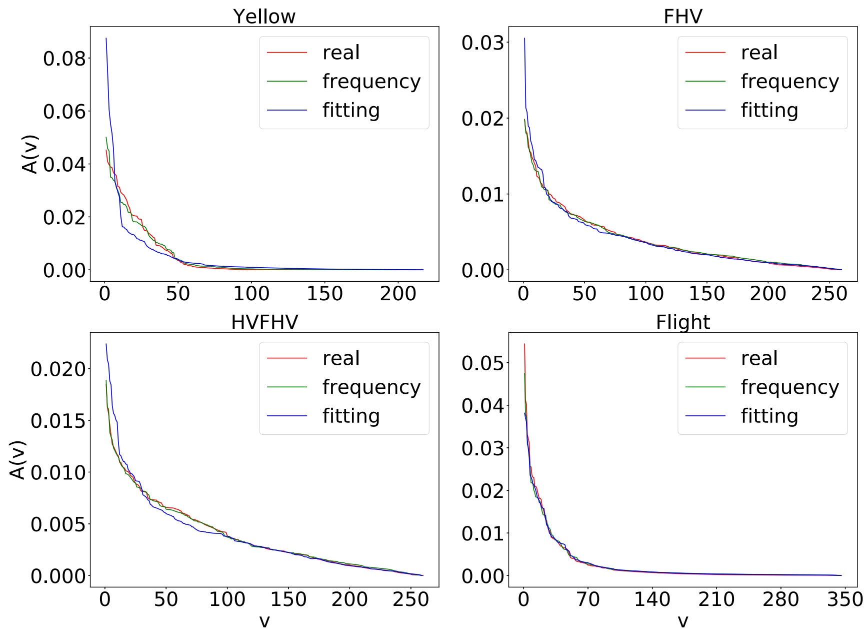

Figure 5 Relative degree of simulation result

even the largest network with around 800K edges could be constructed in less than 15 seconds. This indicates that CDM is highly efficient in network generation. Figures 5, 6, 7, 8, 9 report the measures in the real, frequency-simulation, and fitting-simulation networks. We note that in each subplot, the trend of different curve is similar to each other and the differences are minimal. This indicates that CDM could simulate real networks well.

Scalability of network generation. Next, we investigate the performance of CDM in different categories of networks. Table 6 reports the construction time of various networks in CDM and following results could be drawn from it. First, by varying k, the construction time in the monotonic increases steadily. This is expected because higher k leads to larger $|Active|$ in each competition so that the insert of a newly generated edge becomes more costly. In the non-monotonic, however, the construction time in the non-monotonic sharply decreases at the very beginning and then increases steadily. This is because the low k in the non-monotonic significantly increases the times of invoking

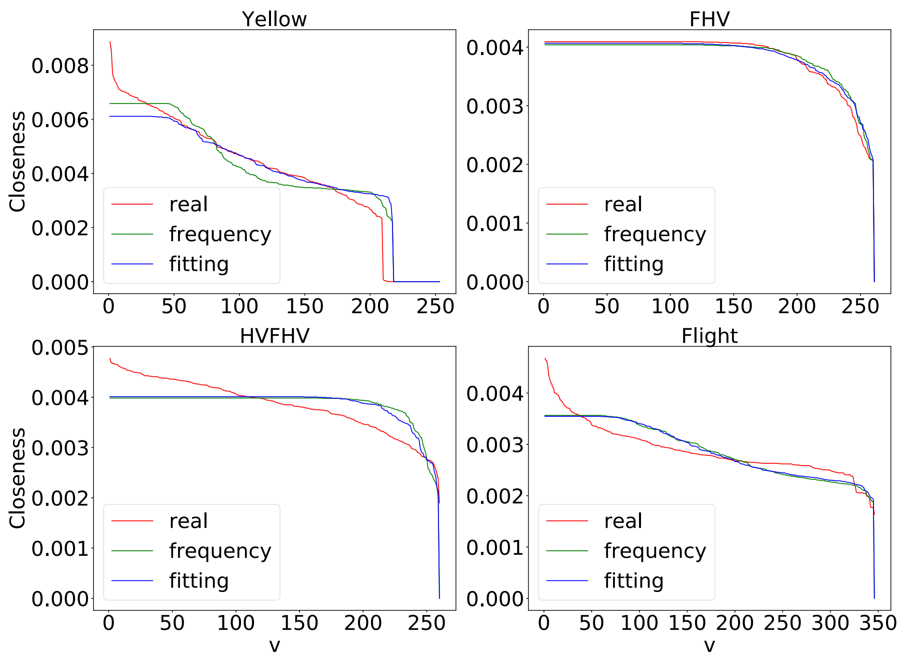

Figure 6 Closeness distribution of simulation result

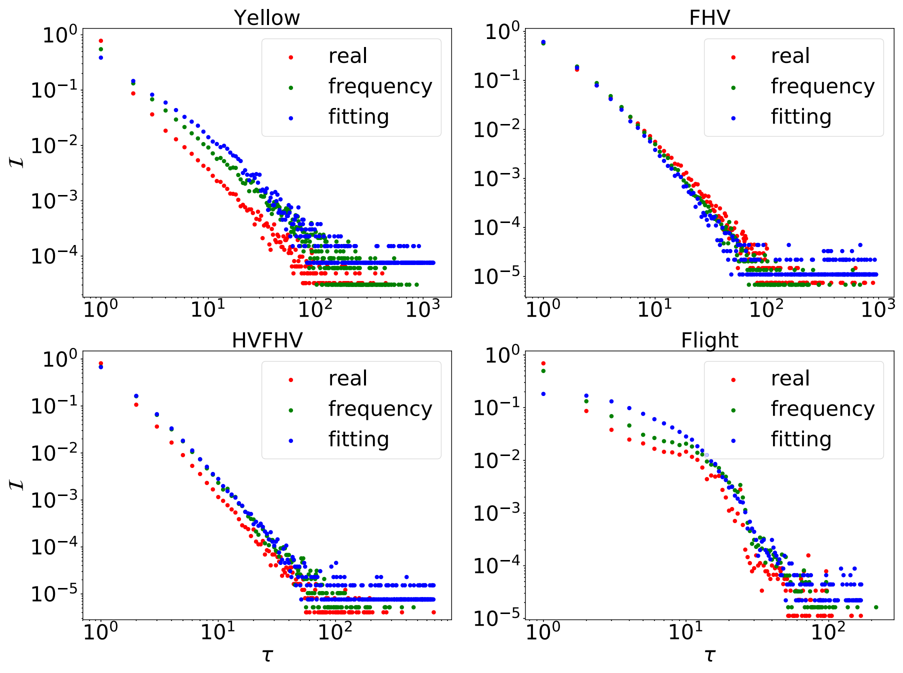

Figure 7 IET distribution of simulation result

PruneActive, which is unproductive to the generation of new edges. Second, by varying $|V|$, the construction time keeps increasing. This is expected because higher $|V|$ generally leads to more participants in a competition which further makes each turn more costly. Third, by varying the desired $|E|$, the construction time steadily increases because

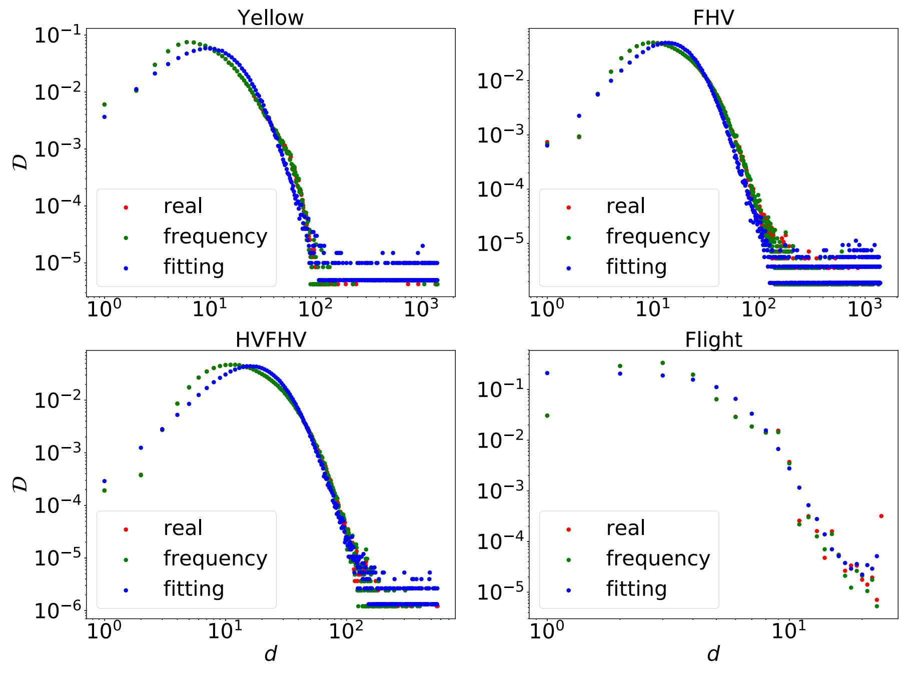

Figure 8 Duration distribution of simulation result

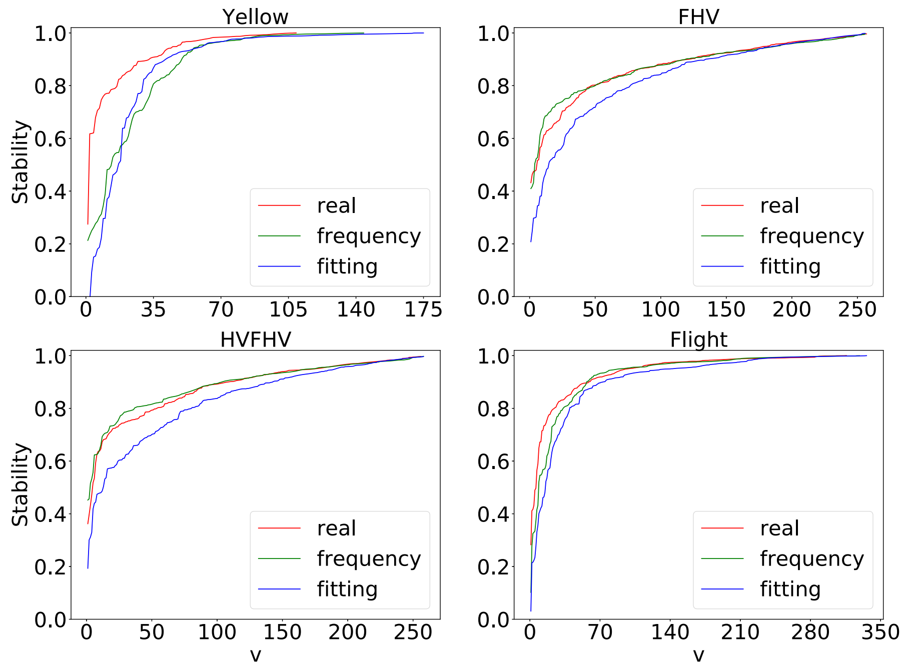

Figure 9 Stability distribution of simulation result

more competitions are carried out to generate larger networks. Fourth, by varying $\overline{\tau}$, the construction time steadily decreases because long IET significantly reduces the number of participants in a competition. Finally, by varying $\overline{d}$, the construction time slightly in-

Table 6 Generation time in both monotonic (denoted t_m) and non-monotonic (denoted t_n) networks with respect to schemas (secs). In most cases, the variation of time is monotonic. The only exception is k-varying t_n, which first decreases and then increases

$k \times 10^{-7}$	1	5	50	500	5000
t_m	865.5	868.7	889.2	892.7	908.6
t_n	2251.8	1010.6	833.6	870.7	921.2

$\|V\|$	500	750	1000	1250	1500
t_m	889.2	887.9	908.6	925.7	920.5
t_n	833.5	867.8	908.6	925.7	948.9

$\|E\|$	200M	400M	600M	800M	1000M
t_m	889.2	1735.4	2680.1	3573.1	4465.7
t_n	833.5	1671.0	2539.8	3390.2	4226.2

$\overline{\tau}$	1	10	100	1000	10,000
t_m	942.2	914.0	878.1	889.2	834.5
t_n	967.0	921.2	873.6	833.5	804.0

$\overline{d}$	1	10	100	1000	10,000
t_m	821.4	810.3	832.2	889.2	881.4
t_n	804.1	819.2	801.0	833.5	840.1

creases because more lasting edges increases the maintenance cost of *Active*. Overall, the result demonstrates that CDM could generate both small and large networks efficiently.

Next, we concentrate on the remaining structural and temporal measures. Figure 10 reports the degree distribution in various networks. Several observations can be made here. First, we note that higher $|V|$ pulls down the degree proportion of each node, which is expected in Equation (19). Second, the higher k lifts the front of the distribution curve while the tail still keeps stable. This is because a higher coefficient value provides more opportunities for higher-power nodes to obtain outgoing edges in each competition so that the high-power nodes could fully take their advantage in their involved competitions. Third, higher $\overline{\tau}$ pulls down the front and lifts the rest of the curve. This is because the appearing of longer inactive period allows lower-power nodes to participate in more competitions without competing with higher-power nodes. Finally, higher $\overline{d}$ pulls down the front part because lasting edges lead to a limited number of edges to be generated at each timestamp. This restricts the degree advantage of high-power nodes.

Figure 11 reports the IET distribution in various networks. The dashed line is there to illustrate the "ends" of the lines, they cannot be seen otherwise because of the significant overlap between the lines that correspond to different studied parameters. We start by drawing two general conclusions about the IET results. First, the configured $\mathcal{I}$ is well modeled in networks generated by the CDM since we can observe the heavy-tails in power-law distribution. Second, the networks with a higher proportion of small IETs tend to have smaller maximal IET. For convenience, we call this the *IET aggregation nature* in the generated networks. Third, the maximal IET in a generated network can be larger than configured $\overline{\tau}$. This is because nodes with lower $\Pi(v)$ may never win in a competition so their IET would be continuously prolonged.

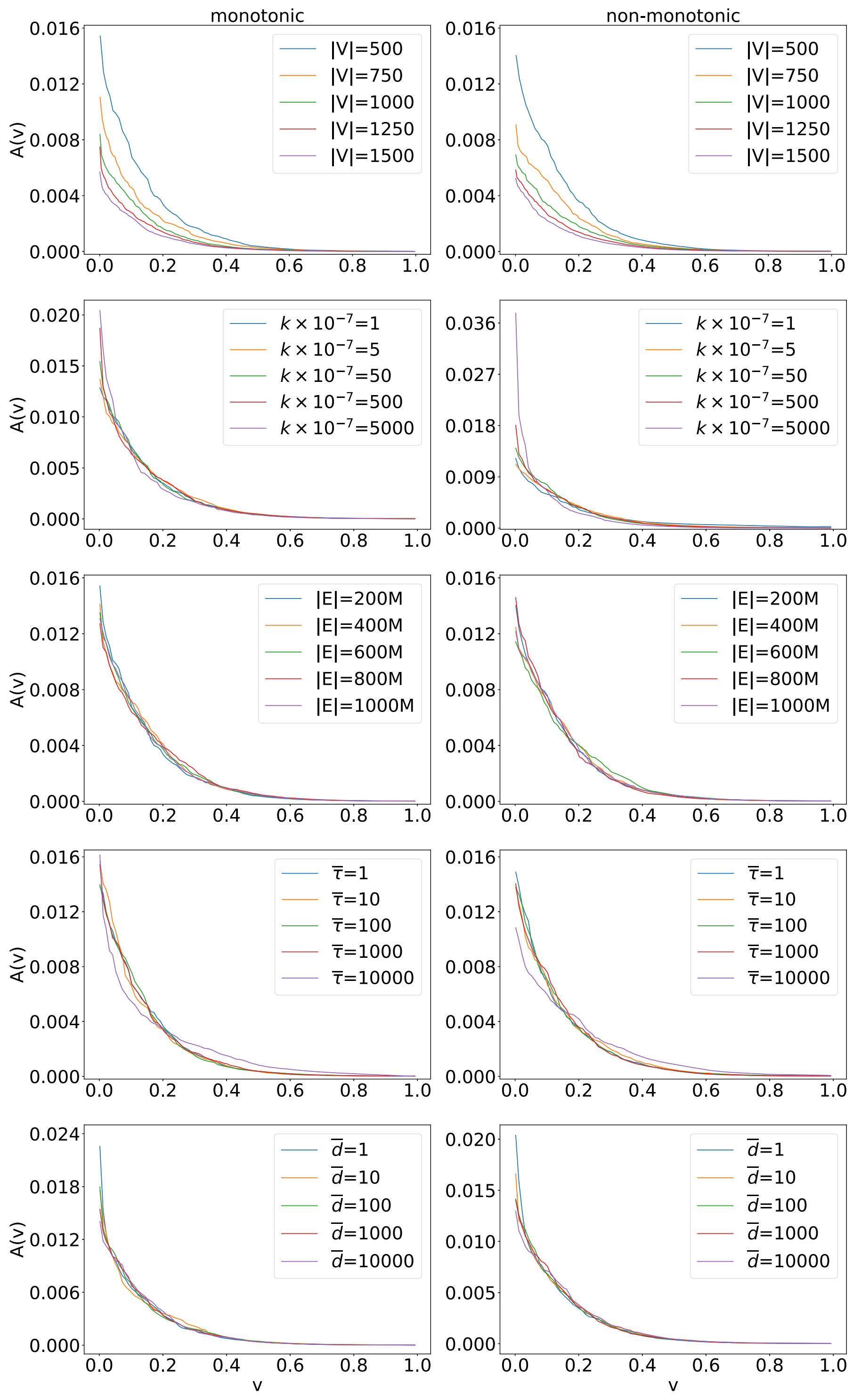

Figure 10 Relative degree in various networks. We note that (1) higher $|V|$, $\overline{d}$ tends to pull down the curve; (2) higher k tends to lifts the curve. (3) higher $\overline{\tau}$ tends to pull down the front and lifts the rest

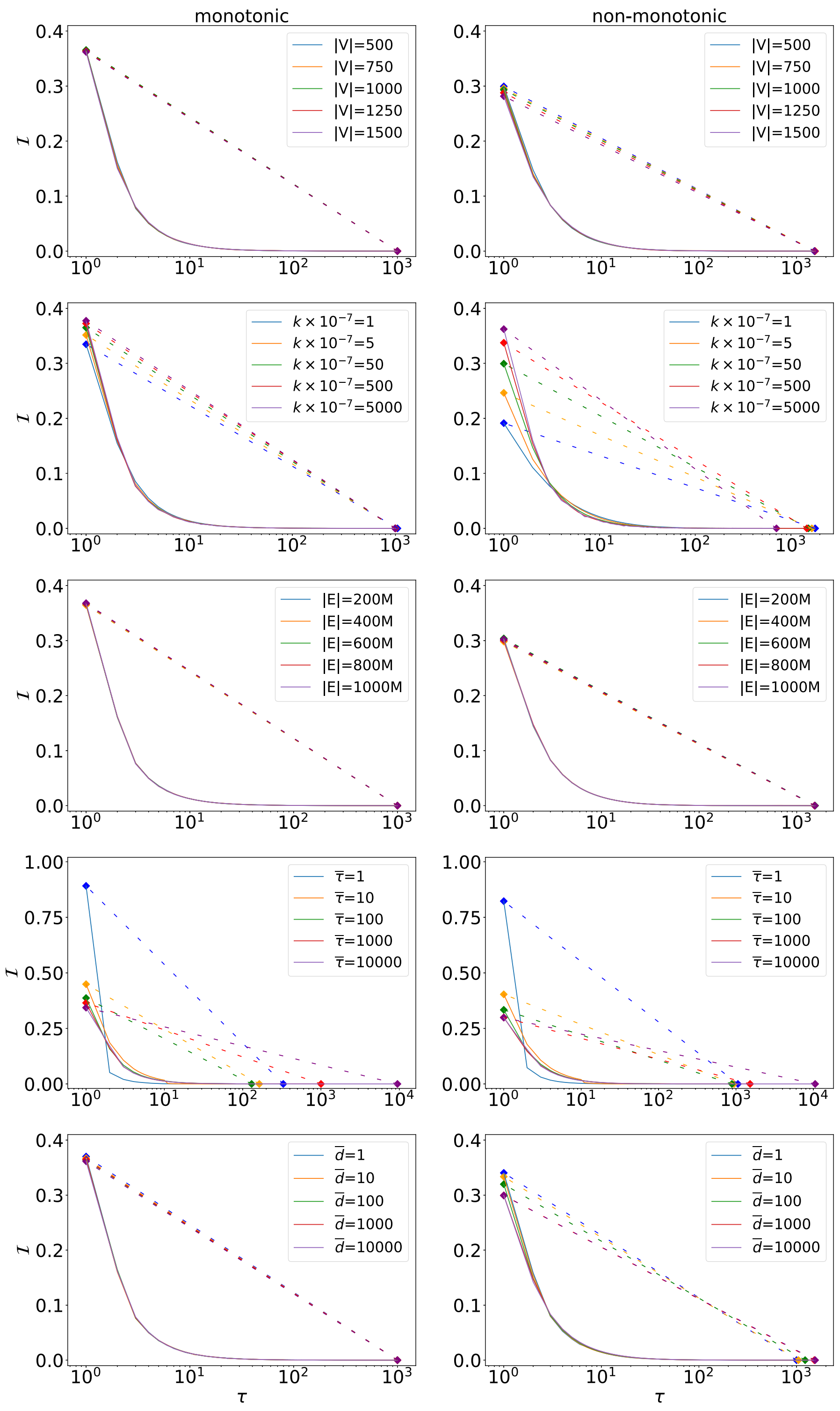

Figure 11 The IET distribution in various networks. We note that (1) higher k tends to enhance the IETs' aggregation one small values; and, (2) higher $\overline{\tau}$, $\overline{d}$ tends to weaken the aggregation

The rest of the results observed in Fig. 11 include the following. First, IET aggregation of the non-monotonic networks tends to be weaker than the monotonic networks. This is because the generation of non-monotonic networks involves the invoking of *PruneActive*, which introduces the period with no competitions and extends nodes' IETs. Second, higher k enhances the IET aggregation because this provides more opportunities for lower-power nodes to win in competitions. Third, higher $\overline{\tau}$ weakens IET aggregation as expected. Finally, higher $\overline{d}$ also weakens the IET aggregation because lasting edges reduce the opportunities for lower-power nodes to win in competitions.

Figure 12 reports the duration distribution in various networks. We first note that the configured $\mathcal{D}$ is also modeled well because of the observed heavy-tails. Second, higher $\overline{d}$ pulls down the durations' aggregation on small values for the similar reason as in IET. Besides, duration proportion in networks generated by the CDM tends to be much more stable since they are hardly impacted by other factors (in comparison to the IET).

Finally, we present the result and analysis of node stability in various networks. A general situation drawn from the resulted statistics is that low-power nodes are generally more stable than higher-power nodes. This is expected because the temporal degree of the low-power is generally small or even negligible comparing to the global maximal degree. To be more specific, considering a node $v \in V$, the global maximal temporal degree might probably vary in v's inactive period. Since small $\Pi(v)$ generally leads to small degree, lower-power nodes tend to be less sensitive to the variation of the global maximal degree. Oppositely, higher $\Pi(v)$ generally leads to in-negligible degree. This makes high-power nodes much much more sensitive to the variation.

Figure 13 reports the node stability in various networks and several results could also be drawn from it: first, the higher $|V|$ lifts the stability curve in both monotonic and non-monotonic networks. It is because the higher $|V|$ leads to the lower degree distribution so that a batch of higher-power nodes become less sensitive and more stable. Second, the higher k lifts the stability curve in both categories because a higher CSS coefficient leads to the increase of the maximal and a higher proportion of stable nodes. Third, higher $\overline{\tau}$ pulls down the curve in both categories because nodes' longer in-active period can intensify the variation of the maximal. Fourth, higher $\overline{d}$ pulls down the curve in both categories because lasting edges can increase nodes' temporal degree and intensify the variation of the maximal. Finally, we note that nodes in the non-monotonic are less stable than that in the monotonic when the rest of the configuration is the same. This demonstrates that *PruneActive*, which mainly considers the generation efficiency and duration distribution in this paper, weakens the stability of nodes in generated networks. So in the future, we would consider various methods used in *PruneActive* and investigate their influence on stability.

6 Conclusion

We proposed the CDM to solve the generation problem of CSS-constrained temporal networks. CDM is designed to take the CSS distribution as an input, which is a vital characteristic of many real networks, as guidance to generate synthetic temporal networks constrained by the CSS. We present theoretical analysis which shows that the cardinality and nodes' relative degrees of a network generated by the CDM could be predicted. Our experimental results also demonstrate that CDM can simulate real networks well and generate networks efficiently.

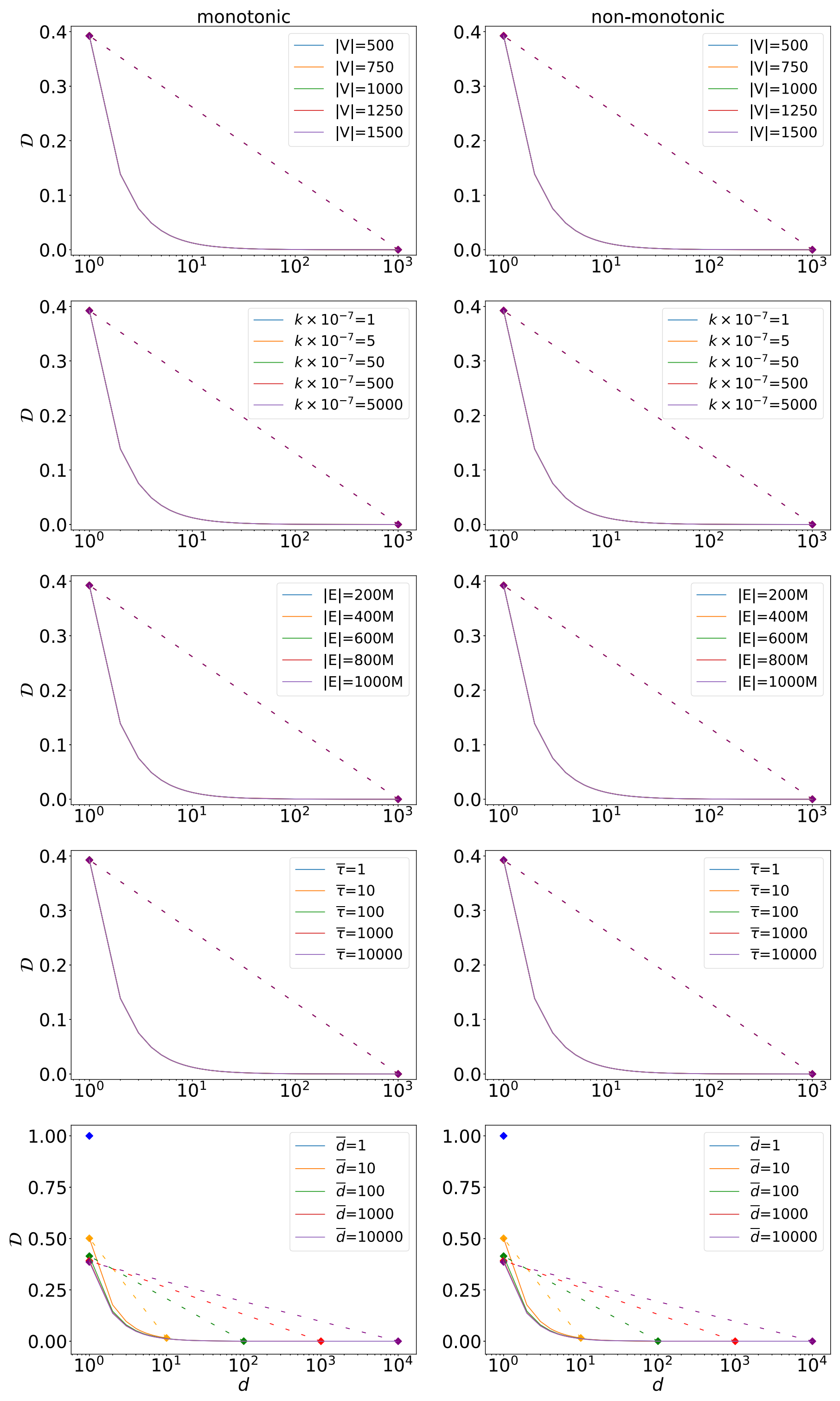

Figure 12 The duration distribution in various networks. We note that higher $\overline{d}$ tends to pull down the durations' aggregation on small values

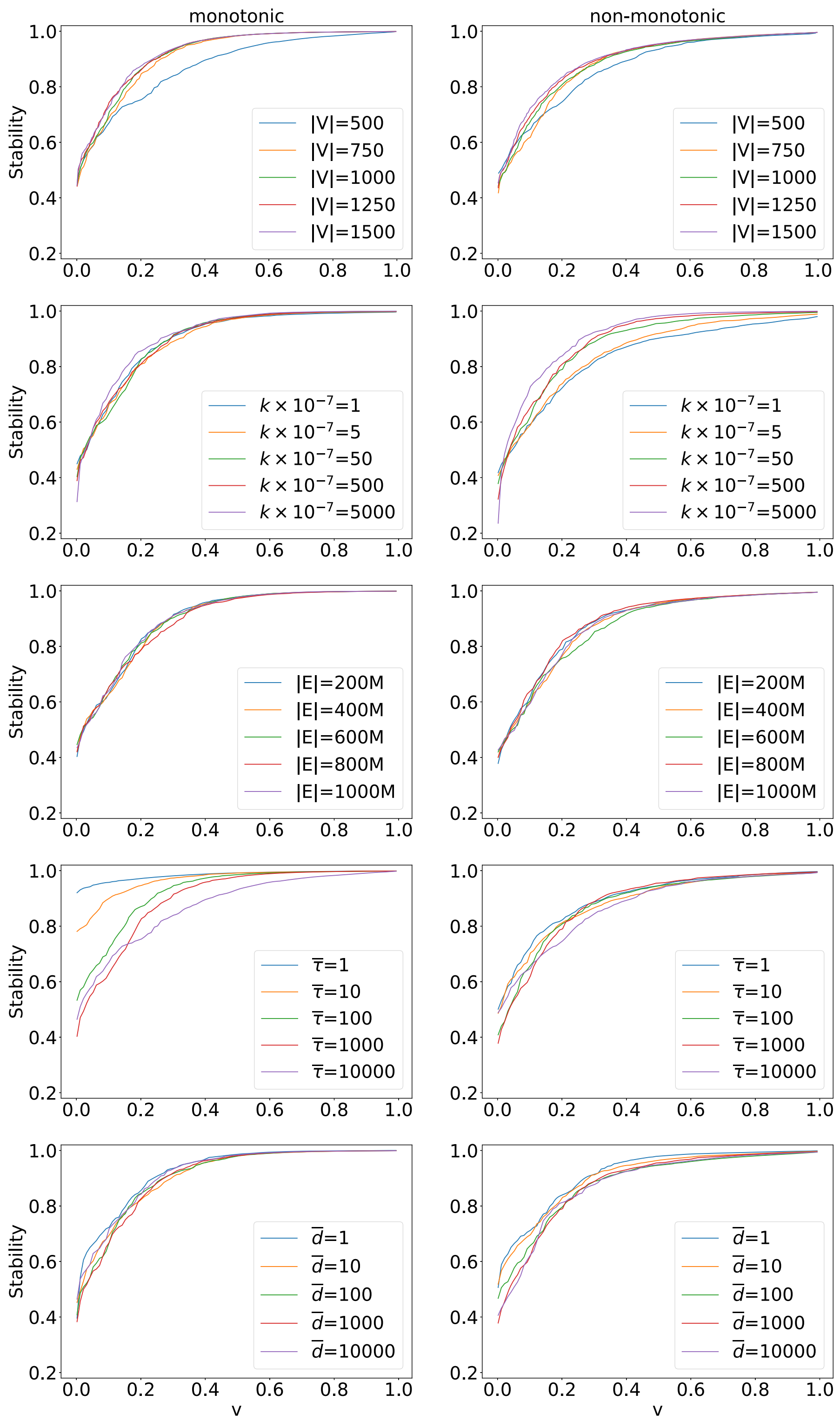

Figure 13 The node stability in various networks. We note that (1) higher $|V|$, k tend to lift the curve; and, (2) higher $\overline{\tau}$, $\overline{d}$ tend to pull down the curve

In future work, we plan to study various *Active*-pruning and destination-selecting strategies and their impact on the networks generated by the CDM.

Acknowledgements
This work was partially funded by the EU H2020 project SmartDataLake (825041), the National Key Research and Development Plan of China (2019QY1300 & 2016YFB0801601).

Abbreviations
CSS, Concurrent set size; ADN, activity-driven networks; IET, inter-event time; CDM, competition-driven model.

Availability of data and materials
The datasets supporting the conclusions of this article are available in the NYC-TLC repository, https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page.

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
All of the authors contributed to designing the proposed model and writing up the results. KZ implemented the model and carried out the analyses and experiments. All authors read and approved the final manuscript.

Publisher's Note

Received: 19 June 2020 Accepted: 26 May 2021 Published online: 03 June 2021

References

1. (2009) NYC Yellow Taxi. https://www1.nyc.gov/assets/tlc/downloads/pdf/data_dictionary_trip_records_yellow.pdf. accessed 15 June 2020
2. (2014) BPI 2014 challenge. https://www.win.tue.nl/bpi/doku.php?id=2014:challenge. accessed 15 June 2020
3. (2015) NYC free hired vehicles. https://www1.nyc.gov/assets/tlc/downloads/pdf/data_dictionary_trip_records_fhv.pdf, accessed 15 June 2020
4. (2019) Airline on-time performance data. https://www.transtats.bts.gov/, accessed 15 June 2020
5. (2019) NYC high-volume free hired vehicles. https://www1.nyc.gov/assets/tlc/downloads/pdf/data_dictionary_trip_records_hvfhs.pdf. accessed 15 June 2020
6. Alessandretti L, Sun K, Baronchelli A, Perra N (2017) Random walks on activity-driven networks with attractiveness. Phys Rev E 95:052318. https://doi.org/10.1103/physreve.95.052318
7. Bagan G, Bonifati A, Ciucanu R, Fletcher GH, Lemay A, Advokaat N (2016) gmark: schema-driven generation of graphs and queries. IEEE Trans Knowl Data Eng 29(4):856–869
8. Boguná M, Lafuerza LF, Toral R, Serrano MÁ (2014) Simulating non-Markovian stochastic processes. Phys Rev E 042:108. https://doi.org/10.1103/physreve.90.042108
9. Bouros P, Mamoulis N (2017) A forward scan based plane sweep algorithm for parallel interval joins. Proc VLDB Endow 10(11):1346–1357. https://doi.org/10.14778/3137628.3137644
10. Cho YS, Galstyan A, Brantingham PJ, Tita G (2014) Latent self-exciting point process model for spatial-temporal networks. Discrete Contin Dyn Syst 19(5):1335–1354. https://doi.org/10.3934/dcdsb.2014.19.1335
11. Gillespie DT (1976) A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. J Comput Phys 22(4):403–434. https://doi.org/10.1016/0021-9991(76)90041-3
12. Holme P (2013) Epidemiologically optimal static networks from temporal network data. PLoS Comput Biol 9(7):e1003142. https://doi.org/10.1371/journal.pcbi.1003142
13. Kim H, Ha M, Jeong H (2015) Scaling properties in time-varying networks with memory. Eur Phys J B 88(12):315. https://doi.org/10.1140/epjb/e2015-60662-7
14. Laurent G, Saramäki J, Karsai M (2015) From calls to communities: a model for time-varying social networks. Eur Phys J B 88(11):301. https://doi.org/10.1140/epjb/e2015-60481-x
15. Masuda N, Rocha LE (2018) A gillespie algorithm for non-Markovian stochastic processes. SIAM Rev 60(1):95–115. https://doi.org/10.1137/16m1055876
16. Nadini M, Sun K, Ubaldi E, Starnini M, Rizzo A, Perra N (2018) Epidemic spreading in modular time-varying networks. Sci Rep 8(1):2352. https://doi.org/10.1038/s41598-018-20908-x
17. Perra N, Gonçalves B, Pastor-Satorras R, Vespignani A (2012) Activity driven modeling of time varying networks. Sci Rep 2:469. https://doi.org/10.1038/srep00469
18. Speidel L, Lambiotte R, Aihara K, Masuda N (2015) Steady state and mean recurrence time for random walks on stochastic temporal networks. Phys Rev E 012:806. https://doi.org/10.1103/physreve.91.012806
19. Starnini M, Baronchelli A, Pastor-Satorras R (2013) Modeling human dynamics of face-to-face interaction networks. Phys Rev Lett 168:701. https://doi.org/10.1103/physrevlett.110.168701
20. Sunny A, Kotnis B, Kuri J (2015) Dynamics of history-dependent epidemics in temporal networks. Phys Rev E 022:811. https://doi.org/10.1103/physreve.92.022811
21. Ubaldi E, Vezzani A, Karsai M, Perra N, Burioni R (2017) Burstiness and tie activation strategies in time-varying social networks. Sci Rep 46:225. https://doi.org/10.1038/srep46225
22. van Leeuwen W, Bonifati A, Fletcher GH, Yakovets N (2017) Stability notions in synthetic graph generation: a preliminary study. In: Proceeding of the 20th international conference on extending database technology (EDBT), Venice, Italy

23. Zhang YQ, Li X, Liang D, Cui J (2015) Characterizing bursts of aggregate pairs with individual Poissonian activity and preferential mobility. IEEE Commun Lett 19(7):1225–1228. https://doi.org/10.1109/lcomm.2015.2437382
24. Zhu K, Fletcher G, Yakovets N (2021) Leveraging temporal and topological selectivities in temporal-clique subgraph query processing. In: Proceedings of the 37th IEEE international conference on data engineering (ICDE). Chania, Crete, Greece
25. Zhu K, Fletcher G, Yakovets N, Papapetrou O, Wu Y (2019) Scalable temporal clique enumeration. In: Proceedings of the 16th international symposium on spatial and temporal databases (SSTD), Vienna, Austria