APPENDIX A

Proof of Theorem 2.1. We reduce a well-known #P-complete problem of counting maximal σ-frequency itemsets [41, 42] to the MFGs counting problem with $\tau_U \geq 1$, $\tau_V \geq 1$ and $\lambda = \sigma$. Let $X = \{x_1, x_2, \ldots, x_n\}$ be a set of items. Given a database $\mathcal{D}$ consisting of $|\mathcal{T}|$ transactions, i.e., $\mathcal{D} = \{d_1, d_2, \ldots, d_{|\mathcal{T}|}\}$. Each transaction d_i comprises several items in X, i.e., $d_i \subseteq X$. The maximal σ-frequency itemsets problem is to enumerate all subsets $\{I_1, I_2, \ldots, I_k, \ldots\}$, where each subset I_k satisfies that *i)* $I_k \subseteq X$, *ii)* $|\{d_i \in \mathcal{D} | I_k \subseteq d_i\}| \geq \sigma$ and *iii)* any superset of I_k does not meet *i)* and *ii)*.

We construct an instance of a temporal bipartite graph $\mathcal{G} = (U, V, \mathcal{E})$ from X and $\mathcal{D}$ to prove the hardness. First, we generate τ_U vertices to form set U. Second, for each item $x_j \in X$, a vertex v_j is correspondingly created to form set V, i.e., $|V| = |X|$. Let $V_i \subseteq V$ be the set of vertices, where each vertex v_j corresponds to each item x_j in the transaction $d_i \in \mathcal{D}$, $|V_i| = |d_i|$. Third, for each transaction d_i, we generate a timestamp t_i such that each $v_j \in V_i$ connects to all vertices in U at this timestamp. Then, there will be $|\mathcal{T}|$ timestamps in $\mathcal{G}$, and (U, V_i) is the only one biclique at the timestamp t_i.

We then show that this transformation of X and $\mathcal{D}$ into $\mathcal{G}$ is a reduction. Suppose $\{I_1, I_2, \ldots, I_k, \ldots\}$ is the set of all the maximal σ-frequency itemsets w.r.t $\mathcal{D}$ in X. We claim that the corresponding vertex sets $\{V_1, V_2, \ldots, V_k, \ldots\}$ is the set of all MFGs in $\mathcal{G}$. Take the vertex set V_k as an example. We first prove that V_k is a λ-frequency group. According to the above construction, V_k can form a biclique with U at no less than λ timestamps, since I_k must be the subset of at least σ transactions in $\mathcal{D}$ and $\lambda = \sigma$. Second, we prove the maximality of the V_k. Suppose to the contrary that V_k is not maximal (i.e., there is a vertex v_j that can join V_k to form a new MFG), I_k is not maximal, since $I_k \cup \{x_j\}$ will be a σ-frequency itemset. This contradicts the condition that I_k is maximal. Third, if there exists an MFG V_z that is not included in $\{V_1, V_2, \ldots, V_k, \ldots\}$, there must exist the other itemset $I_z \notin \{I_1, I_2, \ldots, I_k, \ldots\}$ that is σ-frequency. Therefore, $\{V_1, V_2, \ldots, V_k, \ldots\}$ is the complete set of all MFGs in $\mathcal{G}$. Conversely, assume that $\{V'_1, V'_2, \ldots, V'_k, \ldots\}$ are the set of all MFGs in $\mathcal{G}$. Based on the definition of MFG and the above construction, the one-to-one correspondence between the maximal σ-frequency itemsets of $\mathcal{D}$ and the MFG in $\mathcal{G}$ is established. Therefore, if we count the number of all σ-frequency itemsets of $\mathcal{D}$, we can obtain the number of all MFGs. The reduction is realized. Therefore, the problem of counting the number of MFGs is #P-complete. □

Note that, our enumeration problem is at least as hard as the counting problem, because if we can enumerate all the results, then we can easily count the total number.

Proof of Lemma 3.2. If $|\cap_{v \in V_S \cup \{v'\}} T(v)| < \lambda$, it means that there exists less than λ timestamps, when the number of common m-neighbors of $V_S \cup \{v'\}$ is no less than τ_U. Therefore, $V_S \cup \{v'\}$ is not frequent and we can prune v' from C_V, and the lemma holds. □

Proof of Theorem 3.1. In Algorithm 1, the main procedure for computing the valid candidate set C_V^* is in lines 7-9. We first analyze the time complexity of Algorithm 3. In line 3, the size of U_S is bounded by $d_{max}(v)$ since (U_S, V_S) is a biclique. In lines 5-7, each m-neighbor of u will be visited at most once, which takes $O(d_{max}(u) \cdot |\mathcal{T}|)$ time. The time complexity of updating UA in line 10 is $O(|\mathcal{T}|)$. Similarly, updating λ' also takes $O(|\mathcal{T}|)$ time. Thus, the time complexity of Algorithm 3 is $O(d_{max}(u) \cdot d_{max}(v) \cdot |\mathcal{T}|)$. Moving back to Algorithm 1, every vertex in C_V will be checked by Algorithm 3, and the size of C_V is bounded by the number of vertex in V, i.e., $|V|$. Overall, the time complexity for computing the valid candidate set C_V^* is $O(|V| \cdot d_{max}(u) \cdot d_{max}(v) \cdot |\mathcal{T}|)$. □

Proof of Lemma 3.3. If V_S is an MFG, no vertex from $V \backslash V_S$ can be added into V_S to form a λ-frequency group. The vertices in $V \backslash V_S$ can be categorized as two kinds: one is the non-processed vertex, and the other is the processed vertex. The non-processed vertices that can form a λ-frequency group with V_S will be stored in C_V^* by lines 7-9 of the Algorithm 1. Therefore, $C_V^* \neq \emptyset$ means that there exists the other vertex that can join V_S to form a larger λ-frequency group and V_S is not maximal. The vertex processed before and included in at least one λ-frequency group are stored in X_V. Therefore, if there is a vertex in X_V that can form a λ-frequency group with the current processing vertex set V_S, V_S is the subgraph of one found result and V_S is not maximal. The lemma holds. □

Proof of Lemma 4.1. By extending the antimonotone property of MFG (i.e., Lemma 2.2), if t is not the survived timestamp of V_S, t cannot be the survived timestamp of any superset of V_S. The lemma holds. □

Proof of Theorem 4.1. We first prove that the search branch in line 39 of Algorithm 4 can return all the MFGs containing V'_S that have not been found in the previous search branch. Following the algorithm, any vertex $v' \in cand_V$ satisfies $cnt_T[v'] \geq \lambda$ is the one that can form a λ-frequency group with V'_S, i.e., $V'_S \cup \{v'\}$ is the λ-frequency group. All these vertices can be enumerated during the search and categorized into two kinds, the one whose id is larger than v and the other whose id is smaller than v. If $v' < v$, which means that v' was processed earlier and $V_S \cup \{v, v'\}$ will be enumerated during the search branch for $V_S \cup \{v'\}$. If $v' > v$, v' will be added into C_V^*. It is easy to verify that V'_S is not maximal if $|C_V^*| \neq 0$. Hence, all the MFGs containing V'_S can be enumerated during the branch in line 39. Besides, all vertices in V will be traversed in our algorithm, so that MFG containing each vertex in V can be obtained through the recursion of line 39. Therefore, the theorem is correct. □

Proof of Theorem 4.2. In Algorithm 4, the main procedure of computing the valid candidate set C_V^* are in lines 11-36. The size of survived timestamp set C_T is bounded by $|\mathcal{T}|$ in line 12, i.e., $|C_T| \leq |\mathcal{T}|$. The time complexity of generating $cand_U$ at each timestamp t is $O(d_{max}(v))$ in lines 15-17, and $|cand_U| \leq d_{max}(v)$. Lines 22-30 can be done in constant time, and are performed $d_{max}(u) \cdot d_{max}(v)$ times at each timestamp t. Thus, the time complexity of running lines 12-30 is $O(d_{max}(u) \cdot d_{max}(v) \cdot |\mathcal{T}|)$. In line 32, the size of $cand_V$ is bounded by $d_{max}(u)$. Therefore, Lines 32-36 take $O(d_{max}(u))$ time in the worst case. Overall, the time complexity of computing the valid candidate set C_V^* is $O(d_{max}(u) \cdot d_{max}(v) \cdot |\mathcal{T}|)$. □

APPENDIX B

Algorithms. Note that, BK-ALG cannot finish in a reasonable time if directly applied. Thus, we equip all the algorithms with the graph

Table 3: Statistics of datasets

Dataset	$\|U\|$	$\|V\|$	$\|\mathcal{E}\|$	$\mathcal{E}$ Type	$\|\mathcal{T}\|$	Scale	$(\tau_U, \tau_V, \lambda)$
D1 (MI)	100,000	15,648	58,951	diagnose	25	6month	(6,2,4)
D2 (Ip)	28,540	37,088	73,153	click	31	N/A	(3,2,3)
D3 (diq)	25,771	1,526	133,874	edit	12	year	(3,3,3)
D4 (vec)	33,587	2,282	339,722	edit	14	year	(3,3,3)
D5 (LK)	337,510	42,046	605,642	post	35	year	(3,3,3)
D6 (ben)	249,726	79,269	845,577	edit	17	year	(3,3,3)
D7 (Wut)	530,419	175,215	2,118,877	usage	39	month	(3,2,3)
D8 (Bti)	767,448	204,674	2,517,857	assign	22	year	(3,3,3)
D9 (AR)	1,230,916	2,146,058	5,754,118	rate	21	year	(3,3,3)
D10 (id)	2,183,495	125,482	7,890,901	edit	59	quarter	(3,3,3)
D11 (ar)	2,943,712	209,374	13,601,759	edit	57	quarter	(3,3,3)
D12 (nl)	3,800,350	220,848	28,294,026	edit	65	quarter	(10,6,8)
D13 (it)	4,857,109	343,861	41,146,957	edit	65	quarter	(10,6,8)
D14 (fr)	8,870,763	757,622	66,586,964	edit	66	quarter	(10,6,8)
D15 (de)	5,910,433	1,025,085	70,745,969	edit	67	quarter	(11,11,11)

filtering technique by default. In the experiments, the following algorithms are implemented and evaluated. *i*) **BK-ALG+**: BK-ALG proposed in Section 3 with the graph filtering technique; *ii*) **FilterV**: Algorithm 1 with all the optimizations developed in Section 3; *iii*) **VFree**: Algorithm 4 in Section 4 with graph filtering technique; *iv*) **FilterV-FR**: FilterV without the candidate filtering rule; *v*) **FilterV-VM**: FilterV without the verification methods; *vi*) **FilterV-**: FilterV without both the candidate filtering rule and verification strategies; *vii*) **VFree-**: VFree without graph filtering optimization.

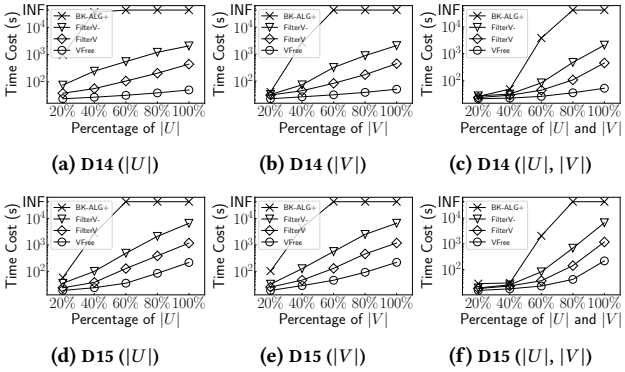
Datasets. We employ 15 real-world temporal bipartite graphs in our experiments, whose details are shown in Table 3. $|\mathcal{T}|$ is the number of snapshots. D1 (MIMIC-III[3]) is a real clinical database that represents relationships between patient and health condition, where the timestamp associated with the edge denotes the time of diagnosis [2, 3, 15]. D2 (Ipvevents[4]) is a real customer-product network, where edges denote the clicking relationships between customers and products. Each relationship between the customer and the product is associated with the label to denote whether the customer is a fraudster or not. The other 13 datasets are obtained from KONECT[5], which are public available. To evaluate the impact of time span, we employ a larger dataset D16 (YS), which is a temporal person–song rating network, with $|U|$=624,962, $|V|$=1,000,991 and $|\mathcal{E}|$=256,804,235 from KONECT.

Parameters and workloads. We conduct experiments by varying parameters τ_U, τ_V and λ, whose default values are shown in the last column of Table 3. For each setting, we run each algorithm 10 times and report the average value. For those experiments that cannot finish within 12 hours, we set them as **INF**. All the programs are implemented in standard C++, and performed on a server with an Intel Xeon 2.1GHz CPU and 64 GB main memory.

APPENDIX C

Scalability evaluation. In this experiment, we use the two largest datasets to demonstrate the scalability of the algorithms. Specifically, for each dataset, we randomly select 20%-80% vertices from

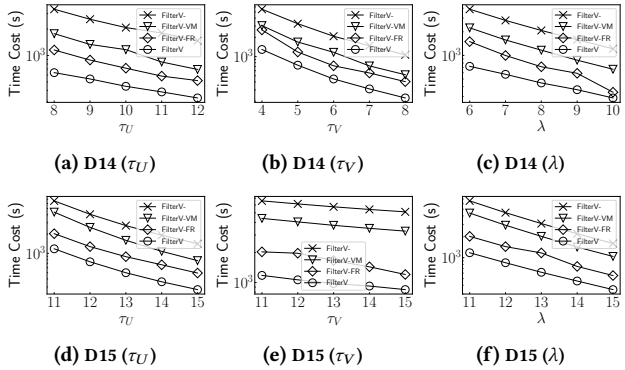
[3]https://physionet.org/content/mimiciii/1.4/
[4]https://tianchi.aliyun.com/dataset/123862
[5]http://konect.cc/networks/



(a) D14 ($|U|$) **(b) D14 ($|V|$)** **(c) D14 ($|U|$, $|V|$)**

(d) D15 ($|U|$) **(e) D15 ($|V|$)** **(f) D15 ($|U|$, $|V|$)**

Figure 12: Scalability testing of all algorithms



(a) D14 (τ_U) **(b) D14 (τ_V)** **(c) D14 (λ)**

(d) D15 (τ_U) **(e) D15 (τ_V)** **(f) D15 (λ)**

Figure 13: Evaluation of the candidate filtering rule and the verification method

U (resp. V) to form four new graphs, and Figure 12(a), 12(d) (resp. Figure 12(b), 12(e)) report the corresponding response time of BK-ALG+, FilterV-, FilterV and VFree. Besides, we randomly select 20%-80% vertices from both U and V to form four new graphs. The response time of these four algorithms are illustrated in Figure 12(c), 12(f) under the default parameter settings. We can find VFree and FilterV outperform the other algorithms and scale well. As the number of vertices in $|U|$ and $|V|$ increases, the response time of all the algorithms grows due to the larger search space. As observed, VFree outperforms the other algorithms by a significant margin and scales well. For example, when the percentage of $|U|$ is 20% in Figure 12(d), the response time of BK-ALG+, FilterV-, FilterV and VFree are 58.38 seconds, 35.74 seconds, 24.05 seconds and 19.15 seconds, respectively. When the percentage of $|U|$ is 80%, BK-ALG+ cannot return results within 12 hours. The response time of the other three algorithms are 2114.26 seconds, 386.98 seconds and 84.90 seconds, respectively. In Figure 12(f), when the percentage of $|U|$ and $|V|$ is 20% in D15, the response time of BK-ALG+, FilterV-, FilterV and VFree are 28.44 seconds, 19.37 seconds, 18.38 seconds and 15.52 seconds, respectively. When the percentage of $|U|$ and $|V|$ is 80%, BK-ALG+ cannot return results within 12 hours. The response time of the other three algorithms are 678.02 seconds, 142.06 seconds and 41.58 seconds, respectively.

Evaluation of the candidate filtering rule and the verification method. In this experiment, we evaluate the performance of the candidate filtering rule (i.e., Lemma 3.2) and the verification method

Table 4: Case study on D2 by varying parameters

$(\tau_U, \tau_V, \lambda)$	**(2,3,4)**	**(2,4,4)**	**(3,2,3)**	**(3,2,4)**
MSG	43.72%	46.15%	39.95%	39.95%
MFB	66.67%	0%	83.33%	50%
MFG	91.67%	100%	92.86%	83.33%

(i.e., Algorithm 3). In Figure 13, We report the response time of FilterV, FilterV-FR, FilterV-VM and FilterV- on D14 and D15 by varying τ_U, τ_V and λ, respectively. The results demonstrate that our filtering rule and validation method can significantly improve the performance of algorithm under all the parameter settings. For example, in D14 with parameters (10,6,8), FilterV-FR, FilterV-VM and FilterV- take 716 seconds, 1166 seconds and 2081 seconds to return the result, respectively. FilterV can return the result within 445 seconds, which verifies the advantage of proposed techniques.

Case study on D2. To evaluate the effectiveness of our model, we conduct a case study on D2 (Ipvevents) to demonstrate the application for anomaly detection. For D2, a vertex denotes a customer (V) or product (U), where each vertex in V is labeled as a malicious (i.e., fraudster) user or not. In e-commerce platforms, members of a fraud group tend to participate in click-farming activities together for various products (targets) frequently. Therefore, analyzing these unilateral frequent vertex sets can facilitate the detection of fraud in e-commerce platforms. We compare our model with MFB and MSG on D2, and report the average accuracy. For each obtained group, we compute the accuracy as the number of malicious vertices divided by the total number of customer vertices, and then take the average value over all the groups. Table 4 shows the accuracy of three models under different parameter settings. As shown, the accuracy of our model is higher than other models and can achieve more than 80% accuracy for all settings. For MFB, although it can achieve over 50% accuracy under some settings, it only returns a few results. For example, MFB returns 66.67% accuracy under parameter setting (2,3,4), but it only returns 1 group, where the number of fraudster vertices is 2 and the total number of customer vertices in the result is only 3.