Top-K High Utility Episode Mining in Complex Event Sequence

Sonam Rathore

Computer Science

Indraprastha Institute of Information Technology, Delhi (IIIT-D), India

A Thesis Report submitted in partial fulfilment for the degree of

MTech Computer Science

1 July 2015

1.: Prof. Vikram Goyal (Thesis Adviser)

2. Prof. Sambuddho Chakravarty (Internal Examiner)

2. Dr. Durvasula Somayajulu (External Examiner)

Day of the defense: 1 July 2015

Signature from Post-Graduate Committee (PGC) Chair:

Abstract

Mining high utility episodes in complex event sequences is an emerging topic in data mining. In utility mining, users set a minimum threshold and the episodes having higher utility than the threshold are reported. The utility framework introduced in episode mining provides more informative and usable knowledge as compared to frequent episode mining. However, it is difficult for the user to set an appropriate minimum utility threshold. As the user cannot predict the count of mined episodes by the utility threshold, the number of reported episodes can vary hugely in accordance to the set threshold. To address this issue, in this thesis we propose an algorithm for mining Top-K high utility episode in complex event sequence. It discovers episodes with highest utility to ones with kth highest utility where the user can set the desired count of episodes, k. We also propose two different strategies to reduce the search space by raising the minimum threshold effectively. We conduct experiments on real dataset and show the effectiveness of our approach.

Acknowledgements

I wish to express my gratitude towards my Thesis mentor Prof.Vikram Goyal whose help, stimulating suggestions and encouragement helped me in all the time of research and writing of this thesis. His excellent guidance and motivation has always been the source of inspiration whenever I got stuck in the work.

I would also like to thank all my thesis committee members : Prof. Sambuddho Chakravarty and Dr. Durvasula Somayajulu for their valuable time.

Finally, I would like to thank my family for their support during the whole period.

Declaration

This is to certify that the MTech Thesis Report titled **Top-K High Utility Episode Mining in Complex Event Sequence** submitted by **Sonam Rathore** for the partial fulfillment of the requirements for the degree of *MTech* in *Computer Science* is a record of the bonafide work carried out by her under my guidance and supervision at Indraprastha Institute of Information Technology, Delhi. This work has not been submitted anywhere else for the reward of any other degree.

Professor Vikram Goyal
Indraprastha Institute of Information Technology, New Delhi

Contents

List of Figures

List of Tables

1

Research Motivation and Aim

Mining of unordered collections of data, transactional and sequential databases, is a popular and important topic of research in data mining. However, there are many applications where the data is in ordered form, e.g., alarms in a telecommunication network, user interface actions, crimes committed by a person, occurrences of recurrent illnesses. Conceptually, the data in these scenarios can be seen as sequence of events, where each event has a time of occurrence and type associated with it. The goal of episode mining is to find relationships between these events.

A lot of studies in this field are dedicated towards episode mining in simple event sequences. Here, the assumption is that only one event occurs at a time point. In Frequent Episode Mining, a typical goal is to find episode with high frequency where the user sets the minimum frequency threshold. According to the set threshold, reported episodes can be very less or huge in number. So, it becomes very difficult for the user to set required threshold in real-life applications. To address this issue, Top-k Frequent Episode Mining was introduced in which episodes with highest frequency to ones with kth highest frequency are mined. The user can precisely control the result of top-k mining by setting the desired number of episodes, k. In this framework, the anti-monotone property is applicable which prunes the search space. Anti-monotone property states that if an episode is infrequent than all of its super episodes are infrequent too. However, the frequent episodes can be of low revenue and the user loses valuable episodes having low frequency. Hence, the FEM cannot satisfy the requirement of users who needs to discover episodes with high utilities like high profits.

The above problem can be solved with the concept of utility mining. The utility of an episode can be defined in terms of weight, profit, cost, quantity or any other measure by the user. Mining High Utility Episodes in Simple Event Sequence refers to finding episodes with utility higher than the user specified minimum utility threshold. However, mining high utility episodes in complex event sequence has more extensive applications than that in a simple sequence. As in complex event sequence, multiple events can appear simultaneously at the same time point. Sequences containing such information are encountered in real-life applications as mentioned by Wu et al.[1]. Like, the purchasing behaviour of a customer can be represented as a complex event sequence. Each item present in the store can be seen as an event having a quantity and a purchase price. All the items bought by the customer in a transaction can be represented at a time point. Mining high utility episodes from such a sequence can find sequential relationships between sets of items. So, mining high utility episodes has emerged as an important topic in data mining.

In this thesis, we propose mining of Top-k high utility episodes from complex event sequence which gives the user required and useable knowledge about the data set. As it is more intuitive and straight forward to set the desired number of episodes than to set the minimum utility threshold, which requires detailed knowledge about the characteristics of data-set.

Discovering top-k high utility episodes in complex event sequences is desirable in many applications but it poses many challenges. First, the framework of utility does not satisfy the monotone or the anti-monotone property. Therefore, it is hard to apply the techniques of top-k frequent mining, which rely on anti-monotonicity to reduce the search space, into top-k utility mining. Second, mining episodes from complex event sequence is much more challenging than mining episodes from simple event sequence. Because in the complex event sequence, more than one type of event can occur simultaneously at the same point of time. Third challenge is how to discover episodes which are potential high utility episodes in the data-set. For discovering potential high utility pattern (abbreviated as PHUI), in high utility pattern mining the concept of TWU (Transaction Weighted Utility) is incorporated. TWU basically represents the upper bound of the patterns utility. A pattern is considered as PHUI if its TWU is no less than the minimum utility threshold. So, to facilitate utility mining of episodes the concept of TWU needs to be incorporated. But it is difficult to change the TWU to

suit episode mining as the data-set mined is a single and long event sequence which is substantially different from the transactional databases and sequence databases used in pattern mining. Fourth, we need to develop strategies to effectively reduce the search space by generating less candidate episodes. As, large number of candidate episodes may increase the execution time and memory consumption.

For mining top-k high utility episodes, motivated from the above, we propose two strategies. These strategies can effectively increase the minimum utility threshold in top-k high utility mining. Also, to prune the huge search space the properties, namely DGE (Discarding Global Events) and DLE (Discarding Local Events), given by Wu et al.[1] are also considered. We observed that though DGE is an effective technique, it cannot be used in top-k mining. Since, the minimum utility threshold is zero in the starting so no event is discarded. But DLE is an effective strategy to prune the search space. The effectiveness of strategies proposed is shown by conducting experiments on real dataset.

Mining this type of data can be useful in many applications. Some of which are mentioned below:

Internet Anomaly Intrusion Detection: Sequences of TCP, UDP, or ICMP network connections can be represented as episodes. Each event in the episode represents an attribute of the traffic connection. For example, the attributes for a TCP connection can be timestamp, duration, service, sourceHost, destinationHost. Analysis of traffic profile using data mining techniques over this type of data helps in distinguishing normal and intrusive traffic.

Biomedical Data Analysis: The biomedical data collected for research can be analysed under episode framework. In Mining episode rules in STULONG dataset [2], the STULONG dataset is mined to find relations between risk factor and clinical demonstration of atherosclerosis. While in Discovering unbounded episodes in sequential data[3], the PROSITE database is used to classify the episodes in event sequences according to the protein family they belong to.

Stock Trend Prediction: In stock market, many factors influence the share price. Each unique factor is considered as an event of different type. A set of events that occur within a short period of time is called an episode. Discovering episodes may help to find out the factors for the fluctuation of prices.

Customer Shopping Sequences: The items purchased by a customer in a transaction can be represented as events occurring at a time point. Hence, mining episodes from such dataset may discover the patterns in the sets of items which further help in analysing the purchasing behaviour of customers.

2

Related Work and Research Contributions

2.1 Related Work

In this section we briefly discuss about some related work done in this field. We divide the work in the following two lines of research:

2.1.1 Frequent Episode Mining

Frequent Episode Mining (FEM) was first introduced by Mannila et al. [4]. Two approaches were proposed in this paper. First approach was WINEPI, in which the events are sampled regularly over a sequence of events. And an episode was considered interesting if it fits into a window width. The window width was defined by the user. The algorithm was based on anti-monotone property of episodes support. The support was calculated by the number of sliding windows in which episode appear. But, it had a defect of duplicate counting of an occurrence of an episode. Hence in second approach, MINEPI, the concept of minimal occurrence was introduced. The minimal occurrence of an episode is a time interval [ts, te] where the episode occurs and it does not occur in any proper sub-interval of [ts, te]. For finding the support, the algorithm counted the number of minimal occurrences of the episode.

The algorithm proposed by Mannila et al.[4], however, generates a large number of candidates to find frequent episodes and hence the computational overhead is significant. To improve the performance of FEM, several methods have been proposed.

Most of the studies are devoted towards mining frequent episodes in simple event sequences[3, 5, 6, 7, 8]. While, discovering frequent episode in complex event sequence is considered only by Huang et al.[9].

Mannila et al. also introduced three classes of episodes. Serial episodes have total order on the events present. Parallel episodes do not have any kind of relative order among events. And the third one was composite episodes which have serial combination of parallel episodes.

Also, there are several ways to define frequency of episodes. Some of them are Window based frequency [4], minimal occurrence[1, 10], non-overlapping[11], non-interleaved occurrence[12]. Achar et al.[6] reviewed many frequency definitions and minimal occurrence based frequency is most commonly used.

2.1.2 Utility Episode Mining

The FEM framework assumes that all the events are of same importance. Therefore, it may report many episodes of low revenue and miss high revenue but low frequency episodes. For solving this issue, the concept of utility is introduced in episode mining by Guo et al.[13]. But this paper only considered the external utility of an episode and mining in simple event sequence. Wu et al.[1] addressed this problem and proposed an algorithm UP-Span to discover high utility episodes in complex event sequence. The events are associated with internal utility (like quantity) and external utility (like profit). It also proposed two strategies, namely Discarding Global unpromising Events; DGE and Discarding Local unpromising Events; DLE, to discard unpromising events and reduce the search space. To speed up UP-Span algorithm, Guo et al. [10], presented a prefix tree structure and tighter upper bounds for candidate episodes utility. From the above related work, we can conclude that only preliminary work is done in mining high utility episodes. Although, mining of high utility patterns from transactional and sequential databases is extensively studied. Many algorithms are proposed for Top-k high utility pattern mining. But, the strategies applied in discovering top-k high utility patterns from transactional database[14, 15] and sequential databases[16] cannot be directly applied to top-k high utility episode mining. As if we transform event sequence to a transactional database or sequential database, than the space requirement is window size time the database size. This makes the methods of mining inefficient.

Also, the frequency of episodes gets defected by transforming the dataset and hence affects the utility of episode.

2.2 Research Contributions

The novel contributions of this thesis are:

1. This thesis presents a new idea of discovering top-k high utility episodes in a complex event sequence.
2. We propose an algorithm to mine top-k high utility episodes.
3. We develop two strategies to make the algorithm efficient by reducing the search space.
4. We conduct experiments on real world data set to show the effectiveness of our approach.

3

Problem Definition and Proposed Solution Approach

3.1 Preliminary

In this section we define some terms and notations. We also denote the top-k utility mining problem.

3.1.1 Episode Mining

Here, we introduce the background for episode mining. We use the notations and definitions given by Wu et al.[1], Guo et al.[10].

Definition 1(Event). An event is defined by the pair (e, t) where e is the event type and t $\in$ N^+ is the time at which event occurs.[1]

Definition 2 (Simultaneous event set). A simultaneous event set is composed of a set of events, where each event occurs at the same time point t.[1]

Definition 3 (Simple event sequence). A simple event sequence S = $\langle(e_1, t_1)$,-$(e_2, t_2), (e_3, t_3), \ldots, (e_n, t_n)\rangle$ is an ordered sequence of events where $t_i<t_j$, for all $1 \leq i<j \leq n$.[1]

Definition 4 (Complex event sequence). A complex event sequence CES = $<(SE_1, t_1), (SE_2, t_2), \ldots, (SE_n, t_n)>$ is an ordered sequence of simultaneous event sets, where each simultaneous event set is associated with a time point t$\in$ N^+ and $t_i<t_j$, for all $1 \leq i<j \leq n$. [1]

Definition 5 (Episode). An episode $\alpha = <(SE_1), (SE_2), \ldots, (SE_n)>$ is a non-empty totally ordered set of simultaneous events, where SE_i appears before SE_j for all $1 \leq i < j \leq n$. [1]

Definition 6 (Occurrence). For an episode = $<(SE_1, t_1), (SE_2, t_2)$,. . . , (SE_n, t_n) >, the time interval $[T_s, T_e]$ is called the occurrence of episode if occurs in $[T_s, T_e]$ and the first simultaneous event SE1 of occurs at time Ts, while the last simultaneous event set SE_k of occurs at time T_e.[1]

Definition7 (Minimal Occurrence). The occurrence time interval $[T_s, T_e]$ of an episode is called a minimal occurrence if there exists no sub-interval of $[T_s, T_e]$ in occurrence of α. Consider two time intervals $[T_s, T_e]$ and $[Ts, Te]$, [Ts, Te]is the sub-interval of [Ts, Te] if Ts Ts and Te Te.[1]

Definition 8 (Support of an episode). The support of an episode is defined as the number of minimal occurrences of an episode.[1]

In mining high utility episodes, each event ei is associated with a positive number p(e_i,CES), called as external utility. Each event e_i in a simultaneous event set SE_j at the time point t_j is associated with a positive number q(e_j, t_j), called as internal utility.

Definition 9 (Utility of an event at a time point). The utility of an event ei at a time point ti is u(e_i, t_i)= p(e_i,CES) q(e_j, t_j). [1]

Definition 10 (Utility of a simultaneous event set at a time point). The utility of a simultaneous event set SE = $\langle (e_1, t_1), (e_2, t_2), (e_3, t_3), \ldots, (e_k, t_k) \rangle$ at a time point t_i is defined as u(SE, t_i) = $\Sigma_{i=1}^{k}$ u(e_j, t_i). [1]

Definition 11 (Utility value of an episode w.r.t. its minimal occurrence). Let mo(α) = $[T_s, T_e]$ be a minimal occurrence of the episode $\alpha = \langle (SE_1), (SE_2), \ldots, (SE_n) \rangle$, where each simultaneous event set $SE_i \in \alpha$ is associated with a time point T_i. The utility of the episode α w.r.t mo(α) is defined as u(α, mo(α)) = $\Sigma_{i=1}^{k}$ u(SE_j, t_i). [1]

Definition 12 (Utility of an episode in a complex event sequence). Let moSet(α) = $[TI_1, TI_2, \ldots, TI_k]$ be the set of all minimal occurrences of the episode α, where TI_i is a minimal occurrence of α for $1 \leq i \leq k$. The utility value of the episode α in a complex event sequence CS is defined as uv(α, CS) = $\Sigma_{i=1}^{k}$ u(α, TI_k). [1]

Definition 13 (High Utility Episode; HUE). If the utility of an episode is not less than the minimum utility threshold, than it is called as a high utility episode. [1]

Definition 14 (Maximum Time Duration). Maximum Time Duration (abbreviated as MTD) is a user specified window. A minimal occurrence mo(α) = $[T_s, T_e]$ satisfies the maximum time duration constraint iff (Te−Ts + 1) ≤ MTD. [1]

Definition 15 (Simultaneous and serial concatenations). Let $\alpha = \langle(SE_1), (SE_2), \ldots, (SE_x)\rangle$ and $\beta = \langle(SE'_1), (SE'_2), \ldots, (SE'_y)\rangle$ be episodes. The simultaneous concatenation of α and β is defined as:
simul-concat$(\alpha, \beta) = \langle(SE_1), (SE_2), \ldots, (SE_x \bigcup SE'_1), (SE'_2), \ldots, (SE'_y)\rangle$.
The serial concatenation of α and β is defined as:
serial-concat$(\alpha, \beta) = \langle(SE_1), (SE_2), \ldots, (SE_x), (SE'_1), (SE'_2), \ldots, (SE'_y)\rangle$. [1]

Definition 16 (Episode-Weighted Utilization of an episode w.r.t a minimal occurrence for a simultaneous concatenation). Let $mo_j(\alpha) = [T_s, T_e]$ be a minimal occurrence of the episode $\alpha = \langle(SE_1), (SE_2), \ldots, (SE_x)\rangle$, where each simultaneous event set $SE_i \in \alpha$ is associated with a time point $T_i (1 \leq i \leq k)$ and $mo_j(\alpha)$ satisfies MTD. The episode-weighted utilization of α w.r.t mo(α) is defined as ESi$(\alpha, moj(\alpha))$= $\Sigma_{i=1}^{k}$ u(SE_i, t_i) + u$(SiC_j(\alpha))$ + u$(SeC_j(\alpha))$.[10]

Definition 17 (Episode-Weighted Utilization of an episode for a simultaneous concatenation). The episode-weighted utilization of α in a complex event sequence CES is defined as ESi (α) = u(α) + u(SiC(α))+ u(SeC(α)).[10]

Definition 18 (Episode-Weighted Utilization of an episode for a serial concatenation). Any Serial concatenation episode of α w.r.t. mo(α) = $[T_s, T_e]$ only considers the event E $\in CES_i | e+1 \leq i \leq s+MTD1$. The episode-weighted utilization of α in a complex event sequence CES is defined as ESe(α) = u(α) + u(SeC(α)).[10]

Definition 19 (High Weighted Utilization Episode; HWUE). An episode is called High Weighted Utilization Episode for serial concatenation or simultaneous concatenation iff its ESe and ESi, respectively, are no less than the minimum utility threshold.[1]

3.1.2 Example

To illustrate some of the previous definition, we take the following example of complex event sequence, CES:

Figure 3.1 shows the occurrence of events in the event sequence and the numbers in brackets show the internal utility of each event at a time point. The Table 3.1 displays the external utility of each of event. An event is defined as ($\langle(A)\rangle$,1)where 1 is the

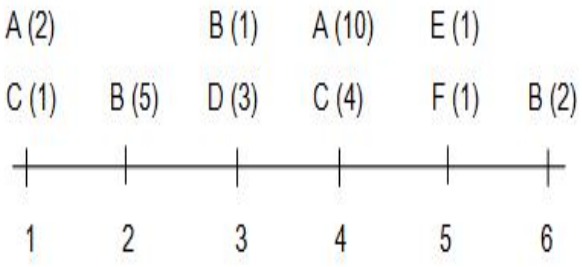

Figure 3.1: A Complex Event Sequence

A	B	C	D	E	F
1	4	2	1	5	10

Table 3.1: External Utility of Events

occurrence time of A. A simultaneous episode can be represented as ($\langle\langle(AC)\rangle$, 1) where (A) and (C) are events occurring at time point 1. The occurrence set of episode (A, B) is occ($\langle\langle(A),(B)\rangle\rangle$) = [1,2], [1,3], [1,6], [4,6]. The minimal occurrences set of episode (A, B) is minOcc($\angle(A),(B)\rangle$)= [1,2], [4,6].

The utility of a single event w.r.t a minimal occurrence can be calculated as u($\langle\langle(B)\rangle$,[2,2])= 54 = 20. While the utility of an event in CES is u($\langle\langle(B)\rangle$, CES)= u(B,[2,2]) + u(B,[3,3])+ u(B,[6,6])= 20+4+8= 32. The utility of a simultaneous event in an event sequence is u($\langle\langle(AC)\rangle$,CES)= [u($\langle\langle(AC)\rangle$, [1,1])] + [u($\langle\langle(AC)\rangle$,[4,4])]= [2+2]+[10+8]=4+18=22. The Utility of an episode u($\langle\langle(A),(B)\rangle$,CES) = u($\langle\langle(A),(B)\rangle$,[1,2]) + u($\langle\langle(A),(B)\rangle$,[4,6])= 22 + 18= 40.

Let the MTD set by user is 3. The SeC($\langle\langle(B)\rangle\rangle$)= [D, A, C, E, F] and SiC($\langle\langle(B)\rangle\rangle$)= [D]. Then, the EWU of an episode for simultaneous concatenation in CES is ESi($\langle\langle(B)\rangle\rangle$) = [u($\langle\langle(B)\rangle\rangle$)]+[u($\langle\langle(D)\rangle$,[3,3])]+[u($\langle\langle(B)\rangle$,[3,3])+u($\langle\langle(D)\rangle$,[3,3])+ u($\langle\langle(A)\rangle$,[4,4])+ u($\langle\langle(C)\rangle$,[4,4]) +u($\langle\langle(E)\rangle$,[5,5]) + u($\langle\langle(F)\rangle$,[5,5])] = [32] + [3]+ [4 + 3+10+ 8+5+10]= 75. While the ESe($\langle\langle(B)\rangle\rangle$) = [u($\langle\langle\langle(B)\rangle\rangle\rangle$)]+[u($\langle\langle\langle(B)\rangle\rangle\rangle$,[3,3])+u($\langle\langle\langle(D)\rangle\rangle\rangle$,[3,3])+u($\langle\langle(A)\rangle$,[4,4])+u($\langle\langle(C)\rangle$,)+ u($\langle\langle(E)\rangle$,[5,5]) + u($\langle\langle(F)\rangle$,[5,5])] = [32] + [4 + 3+10+ 8+5+10] = 72.

3.2 Proposed Solution

In this section, we present an efficient algorithm, TUP (Top-k Utility ePisode mining), for mining top-k high utility episode mining in a complex event sequence. First, we propose a basic algorithm to find the top-k high utility episodes. Then, we propose three strategies to effectively raise the minimum utility threshold.

3.2.1 TUPBasic

This subsection presents an algorithm TUPBasic for discovering top-k high utility episodes. This algorithm does not use a minimum utility threshold; instead list of top-k episodes is maintained dynamically. Top-k List contains k sorted episodes according to their utilities and the minimum utility is raised according to this list.

Pseudo codes in algorithm 1, 2 and 3 are proposed by Wu et al. and Guo et al.[1, 10]. Pseudo code in 1 reads the database and finds all 1-length episodes. The minimal occurrence, utilities and both serial concatenation EWU and simultaneous concatenation EWU are calculated. The EWUs of episodes are compared to the minimum utility which is zero initially but it is raised later in the algorithm. Then, the algorithm explores the search space of high utility episodes with *epi* as prefix.

Pseudo code in 2, simultaneously concatenate events to input episode *epi*. In *Step*2 we find all the vents that can be simultaneously concatenated. In steps $4-9$, a new episode *new_epi* is formed after simultaneous concatenation of *epi* and event e, e $\in$ SiC(epi). Then, the occurrence and minimal occurrence set of the new episode is calculated. In step 18, the utility of *new_epi* is compared with minimum utility and if it greater, then TUPBasic is called to update the Top-k List. If $ESi(new\_epi)$ is not less than minimum utility than SimulConcatenation is called to find high utility episodes with new_epi as prefix. If $ESe(new\_epi)$ is not less than minimum utility than SerialConcatenation is called to find high utility episodes with β as prefix.

Pseudo code in 3 serially concatenate events to input episode *epi*.In *Step*2 we find all the vents that can be simultaneously concatenated. In steps $4-9$, a new episode *new_epi* is formed after serial concatenation of *epi* and event e, e $\in$ SeC(epi). Then, the occurrence and minimal occurrence set of the new episode is calculated. In step 18, the utility of *new_epi* is compared with minimum utility and if it greater, then TUPBasic is called to update the Top-k List. If EWU for Serial concatenation of new_epi, is not less

than minimum utility than SimulConcatenation is called to find high utility episodes with *new_epi* as prefix. If EWU for Serial concatenation of *new_epi*, is not less than minimum utility than SerialConcatenation is called to find high utility episodes with *new_epi* as prefix.

The Steps $11 - 16$ in Algorithm 2 and in Algorithm 3, removes the unpromising events from epi-projected database. This strategy is called as DLE(Discarding Local unpromising Events) suggested by Wu et al.[1]. An event is call unpromising if its EWU is less than the minimum utility threshold. This property holds because of the Episode-Weighted Downward Closure property which states that if EWU of an episode is less than the minimum utility threshold than both simultaneous and serial concatenation with as prefix will give a low utility episode only.

Pseudo code 4 updates the Top-K List and the minimum utility threshold. The input episode *epi* is added to list if the size of list is less than user defined k. Once k candidates are discovered the minimum utility is raised k^{th} highest utility or the least utility in the list. Further, only episodes satisfying minimum utility threshold are inserted in the list and least utility episodes are removed. After the search space is explored, Top-K list contains the desired output of top-k high utility episode mining in complex event sequence.

Algorithm 1: Scan Database(CES, MTD, k)

Data: A complex event sequence CES, maximum time duration MTD, desired number of episodes k

Result: The complete set of top-k high utility episodes

```
Scan CES to find all one length episodes (oneLengthEpiSet).
min_utility=0.
foreach episode epi in oneLengthEpiSet do
    if ESe(epi) ≥ min_utility then
        SerialConcatenation(epi, minOcc(epi), MTD, min_utility, k).
    end
    if ESi(epi) ≥ min_utility then
        SimultConcatenation(epi, minOcc(epi), MTD, min_utility, k).
    end
end
```

Algorithm 2: SimultConcatenation($epi, minOcc(epi), MTD, min\_utility, k$)

Data: an episode epi,minimal occurrence set $minOcc(epi)$,minimum utility $min\_utility$, minimum utility threshold MTD, desired number of episodes k

Result: Simultaneous concatenated episodes with prefix epi

```
foreach [T_s, T_e]inminOcc(epi) do
    SiC_j(epi) = {e, eventoccurringatT_e}.
end
foreach event einSiC_j(epi) do
    new_epi=simult-concatenation(epi, e).
    Occurrence(new_epi) = [T_s, T_e].
    if Occurrence(new_epi) is a minimal occurrence then
        minOcc(new_epi)=minOcc(new_epi) ∪ Occurrence(new_epi).
    end
end
foreach event e ∈ epi projected database do
    if e is an unpromising event then
        remove e from epi projected database.
        remove utility(e) from EWUs of episodes.
    end
end
foreach new_epi do
    if utility(new_epi)>min_utility then
        TUPBasic(new_epi, min_utility, k).
    end
    if ESi(new_epi) ≥ min_utility then
        SimultConcatenation(epi, minOcc(epi), MTD, min_utility, k).
    end
    if ESe(new_epi) ≥ min_utility then
        SerialConcatenation(epi, minOcc(epi), MTD, min_utility, k).
    end
end
```

Algorithm 3: SerialConcatenation($epi, minOcc(epi), MTD, min\_utility, k$)

Data: an episode epi,minimal occurrence set $minOcc(epi)$,minimum utility $min\_utility$, minimum utility threshold MTD, desired number of episodes k

Result: Serial concatenated episodes with prefix epi

foreach $[T_s, T_e] in minOcc(epi)$ **do**
  $SeC_j(epi)$ = e is event occurring at $t \in T_e + 1$ to $T_s + MTD - 1$.
end
foreach event $ein SeC_j(epi)$ **do**
  new_epi=serial-concatenation(epi, e).
  Occurrence(new_epi) = $[T_s, t]$.
  if Occurrence(new_epi) is a minimal occurrence **then**
    minOcc(new_epi) = $minOcc(new\_epi) \cup Occurrence(new\_epi)$.
  end
end
foreach event $e \in epi$ projected database **do**
  if e is an unpromising event **then**
    remove e from epi projected database.
    remove $utility$(e) from EWUs of episodes.
  end
end
foreach new_epi **do**
  if $utility(new\_epi) > min\_utility$ **then**
    TUPBasic($new\_epi, min\_utility, k$).
  end
  if $ESi(new\_epi) \geq min\_utility$ **then**
    SimultConcatenation($epi, minOcc(epi), MTD, min\_utility, k$).
  end
  if $ESe(new\_epi) \geq min\_utility$ **then**
    SerialConcatenation($epi, minOcc(epi), MTD, min\_utility, k$).
  end
end

Algorithm 4: TUPBasic($epi, min\_utility, k$)

Data: an episode epi,minimum utility $min\_utility$, desired number of episodes k
Result: Top-K List: Top-k high utility episodes among the candidates

```
if size(Top − KList)<k then
    Add epi to Top-k List.
end
else
    min_utility ⇐ least utility in Top-K List.
    if utility(epi)>min_utility then
        Remove k^th high utility episode from List.
        Add epi to the List.
    end
end
```

3.2.2 Strategy 1

In this strategy, we sort the episodes w.r.t their EWUs before concatenating them serially or simultaneously with other episodes. So, the order in which the episodes are concatenated is changed.

The utility mining algorithm follows a depth-first search approach to find new episodes. So, a path is searched till its depth and then recursively other paths are called until no more candidates are left. The minimum utility threshold in high utility mining remains fixed and so the path taken does not affect the efficiency. But, in Top-k the order of path taken does matter because the minimum utility threshold is dependent on the candidates list.

This strategy basically identifies the potentially high utility episodes. The episode with highest EWU value is the first candidate to explore the search space. This will help to take the path with highest utility episodes earlier than other paths. Hence, it ensures that the minimum utility threshold is raised effectively.

3.2.3 Strategy 2

This second strategy actually raises the threshold before actually exploring the candidates.

The minimum utility threshold is zero in the TUPBasic which directly affects the performance of the algorithm. While in this strategy the utilities of simultaneous event set present in database are pre-inserted into the Top-k List. When we read database

for the first time, the simultaneous event sets are inserted one by one into the Top-K List. At the end of the database scan the Top-K List contains the top-k simultaneous event sets. So, this strategy raises the minimum utility threshold to a reasonable level before mining, and hence reduces the number of candidates generated.

4

Experimental Analysis and Results

In this section, we show the results of the experiments conducted and evaluate the performance of proposed algorithm. Experiments were performed on a computer with a 2.60 GHz Intel Core 5 Processor with 4 gigabytes of memory running on Windows 8. All the algorithms are implemented in Java language. Real datasets are used to evaluate the performance of algorithms. The types of datasets used are Foodmart, a sparse dataset; Mushroom, a dense dataset; Retail, a large sparse dataset. Though, they are mostly considered as transactional database. They can be considered as a single complex event sequence where each item is an event and each transaction is a simultaneous event set. Foodmart and Retail datasets already consist of internal utility (quantity) and external utility (unit profit) but for mushroom dataset unit profits are generated between 1 to 1,000 by using a log-normal distribution and quantities of items are generated randomly 1 to 5. The average length of the transactions in datasets has been modified.Table **??** shows the statistical information of the datasets used.

Dataset	**Transactions**	**Items**	**Avg. Length**
Foodmart	5581	1559	4
Mushroom	8416	128	10
Retail	88162	16469	5

Table 4.1: Statistical Information of Different datasets

4.0.4 Evaluation on Foodmart Dataset

In this section, we compare the performance of basic algorithm and the algorithm after applying Strategy 1. Figure 4.2 shows the execution time under varied value of k when the minimum utility threshold is set to 5 for Foodmart datset. In figure **??**, the performances of TUPBasic and TUPBasic + Strategy 1 remain equivalent when the value of k is small. But as k increases the Strategy 1 effectively prunes the search space and hence is more efficient than TUPBasic.

In figure 4.4 the effect of variation of MTD value on execution time. The value of K is set to 2. When the value of MTD is large than the performance of Strategy 1 becomes less than TUPBasic and Strategy 2.

4.0.5 Evaluation on Mushroom Dataset

Figure 4.3 shows the number of candidates under varied value of k when the maximum time duration (MTD) is set to 3 for Mushroom dataset. In figure 4.3, the performances of TUPBasic, TUPBasic + Strategy 1 and TUPBasic + Strategy 2 effectiveness to reduce the search space are compared. Strategy 1 is very effective in reducing the number of candidates as compared to TUSBasic. Strategy 2 also reduces the number of candidates but is less effective than Strategy 1.

4.0.6 Evaluation on Retail Dataset

Figure 4.1 represents the performance of the strategies on Retail dataset.It shows execution time under different values of k. The MTD is set as 2. When the value of k is less, the Strategy 1 and Strategy 2 takes almost the same time as the TUPBasic.

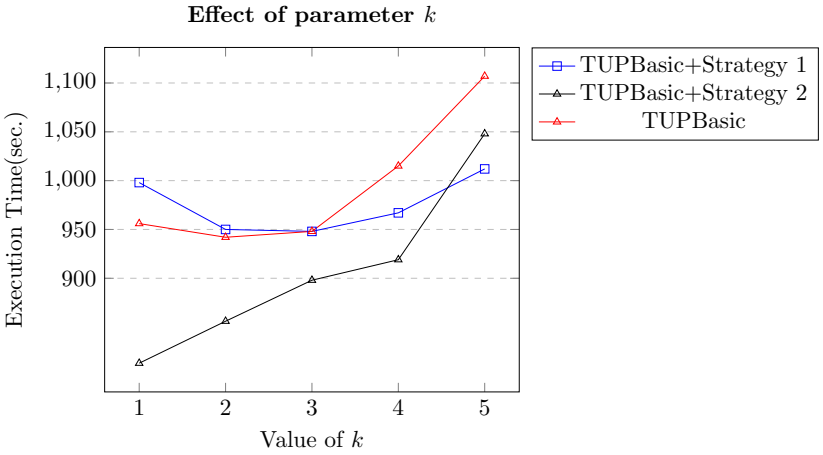

Figure 4.1: The execution time on Retail dataset under different value of k

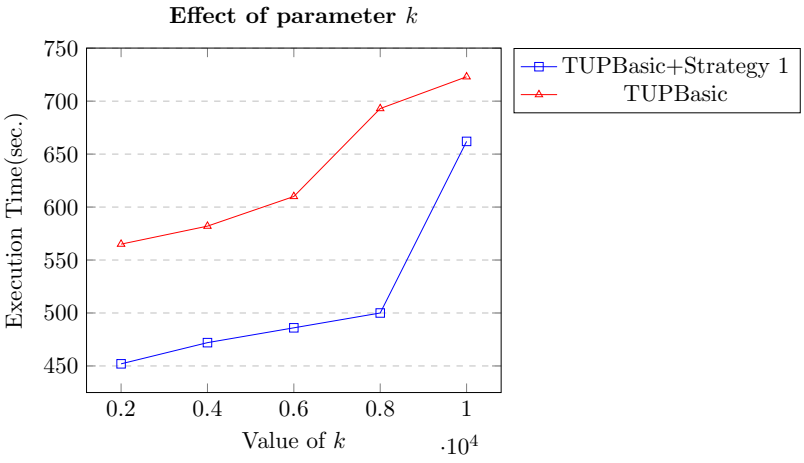

Figure 4.2: The execution time on Foodmart dataset under different value of k

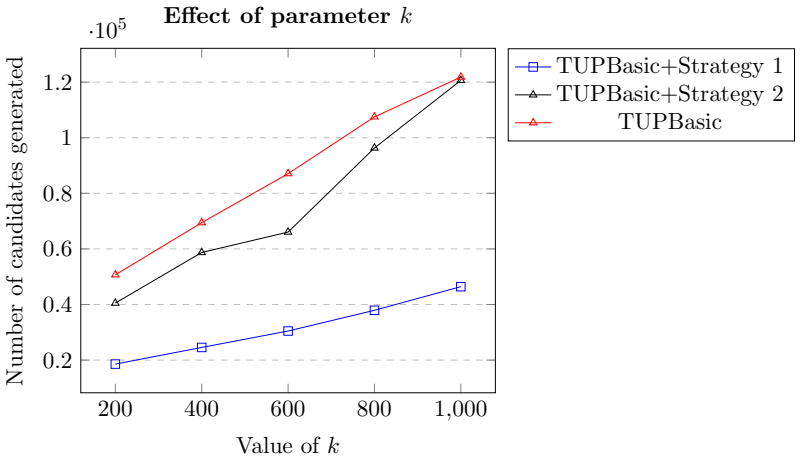

Figure 4.3: The number of candidates on Mushroom dataset under varied value of k

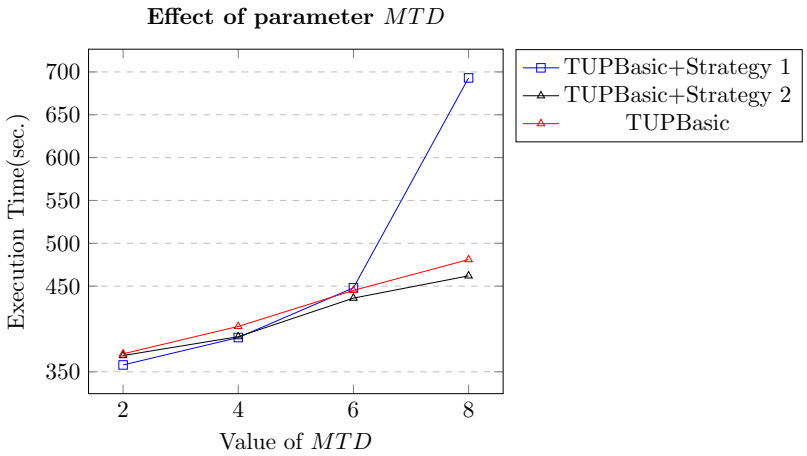

Figure 4.4: The execution time on Foodmart dataset under different value of MTD

5

Conclusion and Future Work

5.1 Conclusion

Utility episode mining in complex event sequences is an emerging topic in data mining and has many applications in real world. In this thesis, we propose an efficient algorithm for mining top-k high utility episodes in complex event sequence. This algorithm guarantees there is no episode missed during the process. Also, we have developed a pruning strategy to reduce the search space. Further, we suggested a pre-insertion strategy to raise the threshold effectively. To evaluate the performance of the proposed algorithm, we conduct experiments on real-world datasets. The results show that the performance is enhanced as the strategies reduced search space and decreased the number of candidates. Though, Strategy 1 is more effective than Strategy 2. Also, Strategy 2 does not give satisfactory results when the number of simultaneous events set is less than the value of k.

5.2 Future Work

We incorporate the concept of top-k utility mining with episode mining in this work. This work considers only serial episodes containing simultaneous events while other kind of episodes like parallel episodes, closed episodes remains to be solved. Also in future work, algorithm improvement and applications can be explored. Further, there are several ways to define the occurrence of episodes such as Window based frequency [4], non-overlapping occurrence [11] which can be addressed in future work. Further in future work, algorithm improvement and applications can be explored.

References

[1] Cheng-Wei Wu, Yu-Feng Lin, Philip S Yu, and Vincent S Tseng. **Mining high utility episodes in complex event sequences**. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 536–544. ACM, 2013. 2, 3, 6, 8, 9, 10, 12, 13

[2] Nicolas Meger, Claire Leschi, Noël Lucas, and Christophe Rigotti. **Mining episode rules in STULONG dataset**. In *In Proc. of ECML/PKDD04 Discovery Challenge-A Collaborative Effort in Knowledge Discovery. Prague: Univ. of Economics*. Citeseer, 2004. 3

[3] Gemma Casas-Garriga. *Discovering unbounded episodes in sequential data*. Springer, 2003. 3, 6

[4] Heikki Mannila, Hannu Toivonen, and A Inkeri Verkamo. **Discovering frequent episodes in sequences Extended abstract**. In *Proceedings the first Conference on Knowledge Discovery and Data Mining*, pages 210–215, 1995. 5, 6, 22

[5] Avinash Achar, A Ibrahim, and PS Sastry. **Pattern-growth based frequent serial episode discovery**. *Data & Knowledge Engineering*, **87**:91–108, 2013. 6

[6] Avinash Achar, Srivatsan Laxman, and PS Sastry. **A unified view of the apriori-based algorithms for frequent episode discovery**. *Knowledge and information systems*, **31**(2):223–250, 2012. 6

[7] Mikhail Atallah, Wojciech Szpankowski, and Robert Gwadera. **Detection of significant sets of episodes in event sequences**. In *Data Mining,*

2004. ICDM'04. Fourth IEEE International Conference on, pages 3–10. IEEE, 2004. 6

[8] Robert Gwadera, Mikhail J Atallah, and Wojciech Szpankowski. **Reliable detection of episodes in event sequences**. *Knowledge and Information Systems*, **7**(4):415–437, 2005. 6

[9] Kuo-Yu Huang and Chia-Hui Chang. **Efficient mining of frequent episodes from complex sequences**. *Information Systems*, **33**(1):96–114, 2008. 6

[10] Guangming Guo, Lei Zhang, Qi Liu, Enhong Chen, Feida Zhu, and Chu Guan. **High Utility Episode Mining Made Practical and Fast**. In *Advanced Data Mining and Applications*, pages 71–84. Springer, 2014. 6, 8, 10, 12

[11] Huisheng Zhu, Peng Wang, Xianmang He, Yujia Li, Wei Wang, and Baile Shi. **Efficient episode mining with minimal and non-overlapping occurrences**. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 1211–1216. IEEE, 2010. 6, 22

[12] Srivatsan Laxman. *Discovering frequent episodes: fast algorithms, connections with HMMs and generalizations*. PhD thesis, Indian Institute of Science Bangalore, 2006. 6

[13] Tianzhu Guo, Shukuan Lin, Ya Wang, and Jianzhong Qiao. **A new framework for detecting high-utility episodes in event sequence**. In *2012 IEEE International Conference on Oxide Materials for Electronic Engineering (OMEE)*, pages 370–373, 2012. 6

[14] Cheng Wei Wu, Bai-En Shie, Vincent S Tseng, and Philip S Yu. **Mining top-K high utility itemsets**. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 78–86. ACM, 2012. 6

[15] Heungmo Ryang and Unil Yun. **Top-k high utility pattern mining with effective threshold raising strategies**. *Knowledge-Based Systems*, **76**:109–126, 2015. 6

[16] Junfu Yin, Zhigang Zheng, Longbing Cao, Yin Song, and Wei Wei. **Efficiently mining top-k high utility sequential patterns**. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*, pages 1259–1264. IEEE, 2013. 6

[17] Min Qin and Kai Hwang. **Frequent episode rules for internet anomaly detection**. In *Network Computing and Applications, 2004.(NCA 2004). Proceedings. Third IEEE International Symposium on*, pages 161–168. IEEE, 2004.