EXCLUSIVE SEQUENTIAL PATTERNS AND THEIR GRAPHICAL REPRESENTATION

Jing Lu[1], Weiru Chen[2] and Malcolm Keech[3]
[1]*School of Computing and Communications - Southampton Solent University, Southampton, UK*
[2]*Faculty of Computer Science and Technology - Shenyang University of Chemical Technology, Shenyang, China*
[3]*Faculty of Creative Arts, Technologies and Science - University of Bedfordshire, Luton, UK*

ABSTRACT

This paper builds on a novel approach to sequential patterns post-processing, where a corresponding data mining method was proposed recently for discovery of Exclusive Sequential Patterns (ESP). A graphical model is introduced as an extension of previous work which utilises exclusive choice and simple merge nodes to realise its ESP-Graph representation. The ESP modelling method includes both a construction algorithm as well as a pairwise refinement and combination phase, to complete the transformation of exclusive sequential patterns to a minimal graphical form. A running worked example describes the process of modelling from the original Sequential Patterns Graph to ESP, illuminating the theories developed and the potential for richer knowledge representation.

KEYWORDS

Exclusive patterns; graphical models; information mining; knowledge discovery

1. INTRODUCTION

Discovering sequential patterns from very large databases is an important application of data mining which has featured in many business and scientific domains, e.g. the analysis of customer behaviour to discover frequent buying patterns; analysis of web access to discover user navigation patterns; prediction of natural disasters; and fraud detection.

The conceptual approach and original mining method were first introduced by Agrawal and Srikant [1995] using a sequence database as input, where each data sequence consists of a list of items or itemsets. Sequential patterns mining aims to find all of the most *frequent* sub-sequences in a given sequence database, i.e. those sub-sequences whose ratios of occurrence exceed a user-specified *minimum support* threshold.

Many algorithms have been introduced in the area of sequential patterns mining over the last decade. Lately Mabroukeh and Ezeife [2010] have investigated the various approaches by introducing a taxonomy for classifying sequential patterns mining algorithms based on the key features supported by these techniques. The proposed taxonomy is composed of three main categories of sequential patterns mining algorithms, i.e. Apriori-based [Agrawal and Srikant 1995; Zaki, 2001], Pattern-Growth [Pei *et al.*, 2001; Yang *et al.*, 2007] and Early-Pruning [Chiu *et al.*, 2004; Song *et al.*, 2005] algorithms. This classification also aims to improve the understanding of sequential patterns mining problems.

The classical sequential patterns mining method returns all frequent sequences present in a database. However, there are often only few which are interesting from a user's point of view. Thus, post-processing is required in order to discard uninteresting sequences. Garriga [2005] addressed the task of summarising sequences by means of local ordering relationships on items. Their work showed that post-processing of closed sequences leads to a generalisation of closed partial orders from sequential patterns. Gomez and Vaisman [2009] proposed the RE-SPaM language to prune sequences resulting from the mining process, using regular expressions.

Structural relation patterns have been introduced to extend the search for more complex patterns hidden behind large sequences of data [Lu *et al.*, 2008]. Discovering these patterns is based on the post-processing of sequential patterns mining results: some sequential patterns may be supported by the same data sequence, and these have been called *concurrent* patterns; while some others may not be supported nor occur together in the same data sequence, and these have been called *exclusive* patterns. Some other sequential patterns may be supported yet occur several times in a data sequence, in such a way that an iterative relationship can be expressed, and this has been called an *iterative* pattern.

Structural relation patterns is thus the general designation of patterns comprising sequential patterns, concurrent patterns, exclusive patterns, iterative patterns and their composition. Structural relation patterns mining has motivated the graph-based modelling of concurrent sequential patterns [Lu *et al.*, 2010], which introduced the ConSP-Graph concept. This paper will in turn focus on the graphical representation of *exclusive* sequential patterns.

Related work is introduced next to provide relevant background on sequential patterns modelling. Following the definition of *exclusion* in patterns mining and corresponding exclusive sequential patterns, ESP, the *Exclusive Sequential Patterns Graph* (ESP-Graph) model is proposed to represent these patterns graphically. The results of ESP mining feed into the associated ESP modelling approach, where a graph construction algorithm and refinement phase are presented with a running worked example for illustration. The paper draws to a close by making brief conclusions.

2. RELATED WORK

The aim of this research is the graph-based modelling of one of the new structural relation patterns from Lu *et al.* [2008], namely exclusive sequential patterns, to inform the analysis of mining results and enhance knowledge representation. This section will describe some related research.

In the field of knowledge discovery and data mining, using graphs is considered to be an expressive and versatile modelling technique providing ways to reason about information implicit in the data. A graph of concepts can be used to model complex relations among data and this can be applied specifically to the modelling of sequential patterns.

Directed acyclic graphs have been used to gain insight into the structure of sequences, where the nodes represent the elements of a sequence and the edges indicate transitions from antecedent elements to subsequent elements. For example, in the work of Joshi *et al.* [2001], a universal formulation has been proposed using graphs to represent inputs to sequential patterns mining. Moreover Loekito *et al.* [2009] demonstrated how a directed acyclic graph, represented as a binary decision diagram, can be used as the data structure for mining frequent sub-sequences.

In sequential patterns mining, given a customer sequence database and user-specified minimum support (*minsup*), the most frequently occurring sub-sequences can be discovered within the database to form a set of sequential patterns. All sequential patterns under the specified minimum support can be generated from the *maximal sequence set*. Thus, a directed acyclic graph called Sequential Patterns Graph (SPG) was defined by Lu *et al.* [2004] to represent the maximal sequence set. Nodes of SPG corresponded to elements in a sequential pattern, i.e. items or itemsets, while directed edges were used to denote the sequence relation between two elements.

For example, consider the sequence database SDB={<*abdac*>, <*eaebcac*>, <*babfaec*>, <*afbacfc*>} and sequential patterns SP={*a*, *b*, *c*, *e*, *f*, *aa*, *ab*, *ac*, *ae*, *af*, *ba*, *bc*, *bf*, *cc*, *ec*, *fa*, *fc*, *aac*, *aba*, *abc*, *abf*, *acc*, *aec*, *afa*, *afc*, *bac*, *baf*, *bcc*, *bfc*, *fac*, *abac*, *abcc*, *abfc*, *afac*, *bafc*} with *minsup*=50%. Figure 1 shows the SPG that models the complete set of sequential patterns.

SPG can be viewed as the visual embodiment of the relationship among sequential patterns. Two special types of node called a *start* node (indicated by double circles) and a *final* node (indicated by a bold circle) were defined to signal the beginning and end of maximal sequences. Any path from a start to a final node corresponds to one maximal sequence. SPG also reflects the minimal representation of a collection of discrete sequential patterns; for example Figure 1 represents all the sequential patterns SP.

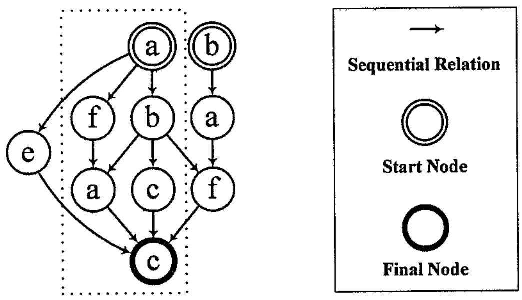


Figure 1. Sequential Patterns Graph example

Consider the segment enclosed within the dotted box above. Not only does it represent maximal sequences *afac*, *abac* and *abcc* but it also represents other sequential patterns *a*, *b*, *c*, *f*, *aa*, *ab*, *ac*, *af*, *ba*, *bc*, *cc*, *fa*, *fc*, *aac*, *aba*, *abc*, *acc*, *afa*, *afc*, *bac*, *bcc* and *fac*. SPG is thus a summary of sequential patterns, useful for presenting results to users.

This SPG modelling approach has been extended to the representation of concurrent sequential patterns [Lu *et al.*, 2010] and will be adapted in this paper to model exclusive sequential patterns, as presented below.

3. EXCLUSION AND PATTERNS REPRESENTATION

Exclusive sequential patterns have been defined in Chen *et al.* [2010], where a corresponding data mining method and algorithms have been presented. It was also pointed out in the previous work that it may be possible to represent exclusive patterns graphically to provide more meaningful information. This section will focus on the concept of exclusion and the formal definition of ESP-Graph.

3.1 Exclusive Patterns

The fundamental concepts related to sequential patterns are covered extensively in the literature [Agrawal and Srikant, 1995; Pei *et al.*, 2001; Zaki, 2001]. For the following definitions, given the sequence database SDB=$\{S_1, S_2,..., S_n\}$; let α, β be two of the sequential patterns mined from SDB with minimum support threshold *minsup* and assume that α, β are not contained in each other, i.e. neither $\alpha\angle\beta$, nor $\beta\angle\alpha$.

With regard to a particular data sequence $S\in$SDB, sequential patterns α and β have an *exclusive* relationship if and only if one of them has occurred in S but not both, i.e. $((\alpha\angle S)\wedge\neg(\beta\angle S))\vee(\neg(\alpha\angle S)\wedge(\beta\angle S))$ is true. It is denoted by $[\alpha–\beta]_S$, where the '–' represents the exclusive relationship.

It may be interesting to detect exclusive relationships within the whole sequence database or within a particular sub-set of SDB, e.g. those sequences where either α or β appear. The degree of exclusion can thus be introduced as follows:

Definition 1. The relative *exclusion* of sequential patterns α and β is defined by

$$exclusion(\alpha,\beta)=|\{S_k:[\alpha–\beta]_{S_k}\}|/|\{S_k:(\alpha\angle S_k)\vee(\beta\angle S_k)\}|$$

where $S_k\in$ SDB and $1\leq k\leq n$.

The denominator in the formula is the number of data sequences which contain either α or β (or both).

Table 1. A sample sequence database and supported SPs

Sequence	Sequential Patterns with *minsup*=50%
<*abdac*>	*a*, *b*, *c*, *aa*, ***ab***, *ac*, *ba*, *bc*, *aac*, *aba*, *abc*, *bac*, *abac*
<*eaebcac*>	*a*, *b*, *c*, *e*, *aa*, ***ab***, *ac*, ***ae***, *ba*, *bc*, *cc*, *ec*, *aac*, *aba*, *abc*, *acc*, *aec*, *bac*, *bcc*, *abac*, *abcc*
<*babfaec*>	*a*, *b*, *c*, *e*, *f*, *aa*, ***ab***, *ac*, ***ae***, ***af***, *ba*, *bc*, *bf*, *ec*, *fa*, *fc*, *aac*, *aba*, *abc*, *abf*, *aec*, *afa*, *afc*, *bac*, *baf*, *bfc*, *fac*, *abac*, *abfc*, *afac*, *bafc*
<*afbacfc*>	*a*, *b*, *c*, *f*, *aa*, ***ab***, *ac*, ***af***, *ba*, *bc*, *bf*, *cc*, *fa*, *fc*, *aac*, *aba*, *abc*, *abf*, *acc*, *afa*, *afc*, *bac*, *baf*, *bcc*, *bfc*, *fac*, *abac*, *abcc*, *abfc*, *afac*, *bafc*

Example 1. (Running example). Table 1 shows the same sequence database SDB={<*abdac*>, <*eaebcac*>, <*babfaec*>, <*afbacfc*>} as above and gives the set of sequential patterns mined with a *minsup* of 50%. This will be used as a running worked example in the paper.

For the sequential patterns pair *ab* and *ae* shown in bold in Table 1, and according to Definition 1, *exclusion*(*ab*,*ae*)=2/4=50%. For another pair *ae* and *af*, also shown in bold, *exclusion*(*ae*,*af*)=2/3≈67%.

The user-specified minimum support threshold *minsup* has been used as the frequency measurement for mining frequent itemsets and sequential patterns. Another value, the minimum *exclusion* threshold *minexc* ($0<minexc\leq 1$), can be introduced to check the exclusive relationship of sequential patterns.

Definition 2. Let *minexc* be the user-specified minimum exclusion. If

$$exclusion(\alpha,\beta)\geq minexc$$

is satisfied, then α and β are called *Exclusive Sequential Patterns* (ESP). This is represented by ESP=$[\alpha–\beta]$ where there is no particular order, i.e. $[\alpha–\beta]=[\beta–\alpha]$.

Therefore, further to Example 1, if *minexc*=65%, then only *ae* and *af* constitute an exclusive sequential pattern given by ESP=[*ae*–*af*]; if *minexc*=50%, there are two such exclusive patterns, i.e. ESP={[*ae*–*af*], [*ab*–*ae*]}.

Problem statement: Given a sequence database SDB=$\{S_1, S_2,\ldots, S_n\}$ and sequential patterns mining results SP=$\{sp_1,sp_2,\ldots,sp_m\}$, i.e. sequential patterns which satisfy a minimum support threshold *minsup*, exclusive sequential patterns mining aims to discover the set of all exclusive patterns beyond a given user-specified minimum exclusion *minexc*.

The following definition was proposed in Chen *et al.* [2010]: an exclusive sequential pattern $ESP_1=[a_1–a_2]$ is *contained* in another exclusive pattern $ESP_2=[b_1–b_2]$ if $((a_1\angle b_1)\wedge(a_2\angle b_2))\vee((a_1\angle b_2)\wedge(a_2\angle b_1))$ is true. This is denoted by $ESP_1\angle ESP_2$. Exclusive sequential patterns are called *maximal* if they are not contained in any other exclusive patterns.

For example, among exclusive sequential patterns [*e*–*f*], [*e*–*af*], [*e*–*abf*], [*e*–*abfc*] and [*aec*–*abfc*], because [*e*–*f*]∠[*e*–*af*]∠[*e*–*abf*]∠[*e*–*abfc*]∠[*aec*–*abfc*], the maximal exclusive pattern is [*aec*–*abfc*]. While this does not imply that all possible combinations of sequential patterns are exclusive in the maximal pattern, it does provide a more compact format with potential for knowledge discovery.

Example 2. According to the ESP mining method and component algorithms [Chen *et al.*, 2010], with its focus on *pairwise exclusion*, a set of just six maximal exclusive sequential patterns can be mined from SDB with *minexc*=65% in Example 1: ESP_6={[*aec*–*abcc*], [*aec*–*abfc*], [*aec*–*afac*], [*aec*–*bafc*], [*abcc*–*abfc*], [*abcc*–*afac*]}.

Exclusive relationships may be extended beyond pairs of patterns ESP=$[\alpha–\beta]$ to three sequential patterns α, β and γ and, more generally, to a set of patterns $\{sp_1,sp_2,\ldots,sp_r\}$. This is defined in Chen *et al.* [2010] and represented exclusively as $[sp_1–sp_2–\ldots–sp_r]$.

3.2 ESP-Graph Representation

The use of graph-based modelling in data mining has led to the development of a sequential patterns model that explores the inherent relationship among sequential patterns. The idea has been applied to concurrent sequential patterns [Lu *et al.*, 2010] and it is adapted here for modelling exclusive sequential patterns. The definition of SPG [Lu *et al.*, 2004] is thus extended to introduce Exclusive Sequential Patterns Graph and followed by an example for illustration.

Definition 3. *Exclusive Sequential Patterns Graph* (ESP-Graph) is a graphical representation of exclusive sequential patterns, ESP denoted by a 7-tuple expressed as follows: ESP-Graph=(V, E, S, F, C, M, δ), where

(1) V is a non-empty set of nodes. Each item (or itemset) in ESP corresponds to one node in ESP-Graph and each node in ESP-Graph at least corresponds to one item in ESP.

(2) E is a set of directed edges. The sequential relation of any two adjacent items in a sequence of ESP corresponds to the directed edge of two nodes in ESP-Graph. Any one directed edge at least corresponds to the sequential relation of two adjacent items in a sequence of ESP.

(3) S is a set of start nodes, $S\subseteq V$, and $S\neq\varnothing$. There are no start nodes which have the same value in any individual ESP-Graph.

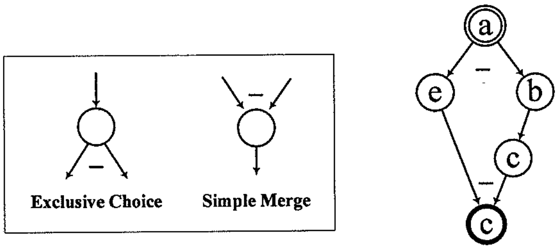
(4) F is a set of final nodes, $F\subseteq V$, and $F\neq\varnothing$. There are no final nodes which have the same value in any individual ESP-Graph.

(5) C is a set of *exclusive choice* nodes, C⊆V, allowing independent execution between exclusive paths, modelled by connecting two or more outgoing sequential relations.

(6) M is a set of *simple merge* nodes, M⊆V, with two or more incoming sequential relations applied to exclusive paths to allow no more than one outgoing sequential relation.

(7) δ is a function from a set of directed edges to a set of pairs of nodes. δ can also be defined as a map function of V→V, which indicates the relations between any two nodes.

For any node in ESP-Graph, neither the subsequent paths can be the same nor the antecedent paths. If two nodes have the same value, they must have different antecedent and subsequent paths. The additional graphical elements used in relation to ESP-Graph are derived from workflow control [van der Aalst *et al.*, 2003] and shown in Figure 2(a), where the '–' represents the exclusive relationship across connected paths.



(a) (b)

Figure 2. ESP-Graph elements and sample representation

It is worth noting that a '+' denotes *concurrent* relationships in the corresponding ConSP-Graph models [Lu *et al.*, 2010].

Example 3. The exclusive sequential pattern [*aec–abcc*] from Example 2 can prospectively be cast into an equivalent graphical representation, as in Figure 2(b).

Node *a* inside the double circles is not only a start node but also an exclusive choice node, connecting two outgoing sequential relations *ec* and *bcc*. And node *c* inside the bold circle is not only a final node but also a simple merge node, with two incoming sequential relations *ae* and *abc*. The methodology for construction of ESP-Graphs is discussed in the next section.

4. EXCLUSIVE SEQUENTIAL PATTERNS MODELLING

As with concurrent sequential patterns, the natural way to consider transforming exclusive sequential patterns into a graphical representation, ESP-Graph is by identifying the inherent relationships through common prefix/postfix recognition. This section discusses the methods for modelling pairwise exclusive sequential patterns, constructing and then refining graphs when suitable combinations exist. While the running worked example illustrates maximal exclusive patterns, the approach can be used to represent all ESPs.

4.1 ESP-Graph Construction

The definition of ESP-Graph is an extension of that for sequential patterns graph and therefore the SPG construction algorithm can be adapted for ESP modelling, as specified below.

1. *Initialisation.* Given a set of *n* pairwise exclusive patterns $ESP_n=\{[sp_{i1}-sp_{i2}] | 1 \le i \le n\}$. Let $i=1$.

2. *Construction.* For each exclusive sequential patterns pair $[sp_{i1}-sp_{i2}]$ ($1 \le i \le n$), represent sp_{i1} by a directed graph G_i; find any common prefix and/or postfix shared between sp_{i1} and sp_{i2}. If a common prefix/postfix exists, then use the *construction algorithm* to generate the next transitional graph model G_i; otherwise represent sp_{i2} by a directed graph G' and set $G_i=G_i \cup G'$.

3. *Iteration.* For $i=i+1$, repeat step 2 with the pair $[sp_{i1}-sp_{i2}]$ from ESP_n until $i=n+1$. The final result $G=G_1 \cup G_2 \cup \ldots G_n$ is the union of the ESP-Graphs.

The algorithm below shows the pseudo-code for the ESP-Graph construction phase following an original approach for SPG modelling adapted by Lu *et al.* [2010].

ISBN: 978-972-8939-53-3 © 2011 IADIS

ESP-Graph Construction Algorithm
Input: An exclusive sequential patterns pair [sp_{i1}–sp_{i2}] and a transitional graph model G_i
Output: A new directed graph G_i following incremental construction
Procedure:
preS=common prefix of sp_{i1} and sp_{i2}
postS=common postfix of sp_{i1} and sp_{i2}
elemS=sp_{i2}-*preS*-*postS*
Represent *elemS* by the directed graph G'
If *preS* is not empty
{The last node of *preS* in G_i is an exclusive choice node;
Add a directed edge from it to the first node of G';
Mark the connected paths with a '–'}
If *postS* is not empty
{The first node of *postS* in G_i is a simple merge node;
Add a directed edge to it from the last node of G';
Mark the connected paths with a '–'}
This new directed graph models [sp_{i1}–sp_{i2}] and is called G_i.

Example 4. The modelling of the set of exclusive sequential patterns from Example 2 using ESP-Graph construction can be described as follows:

1. *Initialisation.* Given ESP_6={[*aec–abcc*], [*aec–abfc*], [*aec–afac*], [*aec–bafc*], [*abcc–abfc*], [*abcc–afac*]}.
2. *Construction.* For the first exclusive sequential patterns pair [*aec–abcc*], represent *aec* by a directed graph G_1; both a common prefix *preS=a* and *postfix* postS=*c* can be found. Taking out *preS* and *postS* from *abcc*, the remaining part *elemS=bc* can be represented by a graph G'. Add a directed edge from the last node of *preS* in G_1 (i.e. *a*) to the first node of G' (i.e. *b*); *a* is an exclusive choice node; mark the connected paths with '–'. In the same way, final node *c* is a simple merge node and connected paths can be marked with '–'. The result is graph G_1 shown in Figure 3, where dotted lines represent new edges in the transitional model.
3. *Iteration.* ESP-Graphs can be generated from each of the exclusive sequential patterns pairs by iteration, using the construction algorithm above. Figure 3 gives the complete graphical representation at this first stage of ESP modelling.

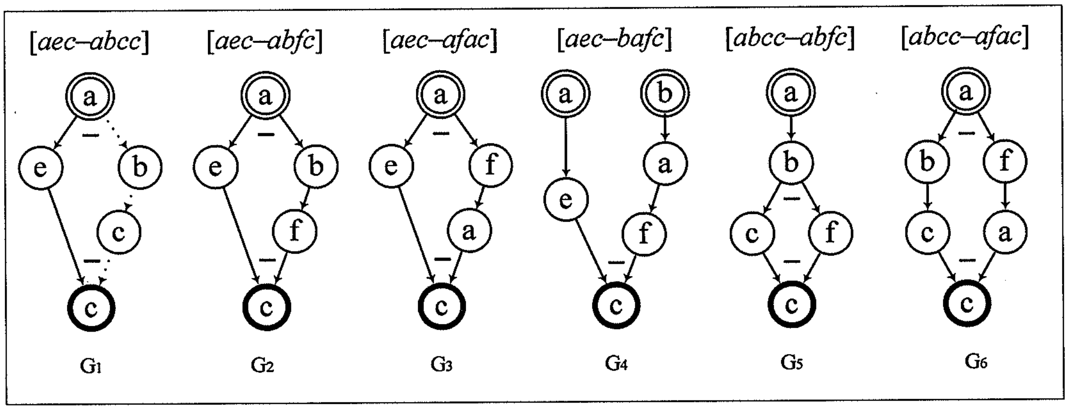


Figure 3. Modelling of exclusive sequential patterns

4.2 Pairwise Refinement and Combination

The running example shows that extending the SPG modelling methodology to ESP-Graph construction is straightforward in practice. It can be seen that the graphs in Figure 3 bring together both connectivity and structure. In addition, G_1, G_2, G_3 and G_4 all share the exclusive path *aec*, which indicates the potential for enhancing the pattern representation through a suitable approach.

There are five steps in the following method for pairwise refinement and combination of exclusive sequential patterns graphs.

1. *Initialisation.* Given the set of ESP-Graphs G for ESP_n. Let G'=G, i=1 and j=i+1.

2. *Refinement.* For a given i, j and pairs of G_i and G_j in G, where $1 \le i < n-1$, $i < j < n$, refine the model by finding each occurrence of a common exclusive path – if a pair of graphs share such a path G" from a start node to a final node, then go to step 3 – otherwise go to step 4.

3. *Combination.* If (G_i-G") and (G_j-G") constitute an existing graph G_k ($j < k \le n$) within G, then G_i, G_j and G_k contain mutually exclusive sequential patterns – combine G" and G_k, using a variation of the *ESP-Graph construction algorithm*, to generate the new graph G_{ij} – accumulate this graph in the transitional model G'.

4. *Iteration.* Continue this cycle from step 2 with j=j+1 through each remaining pair of graphs in G, until j=n. Let i=i+1, j=i+1 and return to step 2 unless i=n-1.

5. *Deletion.* Delete all the graphs from G' which have been used successfully for combining into new graphs, at which point G=G' is the new ESP-Graph model.

Example 5. For the exclusive sequential patterns graphs from Example 4, the following illustrates the procedure of refinement and combination using the above method.

1. *Initialisation.* Given the ESP-Graphs G from Figure 3.

2. *Refinement.* For the first pair in G, they share an exclusive path *aec* from start node *a* to final node *c*.

3. *Combination.* (G_1-*aec*) and (G_2-*aec*), i.e. *abcc* and *abfc*, make up another graph G_5 from Figure 3, so G_1, G_2 and G_5 contain mutually exclusive sequential patterns. These graphs combine to generate a new graph G_{12}, as shown in Figure 4. Accumulate this new graph in the transitional model G'.

4. *Iteration.* Continue this cycle from step 2 and similarly generate the new graph G_{13} from G_1, G_3 and G_6 (as above). The next pair G_1 and G_4 also share the exclusive path *aec*, but the combination of (G_1-*aec*) and (G_4-*aec*) does not feature as an ESP-Graph in G. Iterations proceed, but no further refinement takes place.

5. *Deletion.* Delete graphs G_1, G_2, G_3, G_5, and G_6 from G' to form the new overall model G in Figure 4.

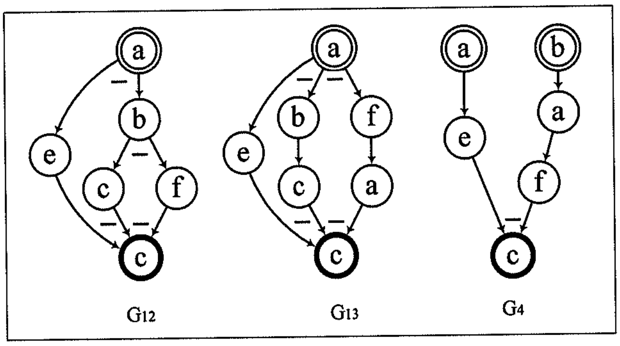


Figure 4. Refinement and combination of ESP-Graphs

Cross-referencing with the SPG model from Figure 1, which graphically represents maximal sequential patterns, Figure 4 explores further the relationships between these patterns in the context of exclusion. For example, graph G_{13} represents an *exclusive choice* from the start node *a*, allowing independent execution among the three mutually exclusive paths. These paths then feature as incoming sequential relations through *simple merge* to the final node *c*. Thus knowledge representation is more explicit in the ESP-Graph model.

Moreover, further application of the above method could generate even higher-level graphs in principle, if another round of pairwise refinement and combination was pursued. Following the running example, G_{12} and G_{13} exhibit sufficient commonality in Figure 4 that – if an appropriate "G_7" existed, which combined *abfc* and *afac* – then further refinement of ESP-Graphs could proceed as above to form a corresponding G_{123} as part of a new model. Alas, the final ESP-Graph model remains as in Figure 4 in this case.

5. CONCLUSIONS

The data mining method of discovering exclusive sequential patterns has been taken to the data modelling phase in this article, with the introduction and demonstration of the ESP-Graph representation. Algorithms are given for original graph construction as well as the pairwise refinement and combination of ESP-Graphs, yielding interesting results and potential for further work. In particular, extension of the theory to cater for ESP=[α–β–γ] would enable more complex modelling of relationships beyond just pairwise exclusion.

There are natural applications for ESP mining and modelling not only in workflow design but also customer transaction analysis and prediction. For example, using sequential patterns mining to extract customers' purchasing behaviour from transaction records and applying exclusive sequential patterns mining enables a more detailed analysis of their buying or non-buying patterns. This could help in design of marketing strategies (e.g.) to avoid some types of unlikely cross-selling. The same method may be applied to discover users' browsing behaviour from web access logs.

ESP mining and modelling could be developed from pairwise to multiple sequential patterns, although the level of mutual exclusion required across all combinations becomes more challenging. The relationship between *minexc* and *minsup* would appear to be integral to the discovery of the most useful exclusive patterns, which is another research direction to be investigated.

REFERENCES

Agrawal, R. and Srikant, R., 1995. Mining sequential patterns. *Proceedings of 11th International Conference on Data Engineering*. Taipei, Taiwan: IEEE Computer Society Press, pp. 3-14.

Chen, W.R., Lu, J. and Keech, M., 2010. Discovering exclusive patterns in frequent sequences. *International Journal of Data Mining, Modelling and Management*, 2(3), pp. 252-267.

Chiu, D.Y., Wu, Y.H. and Chen, A.L.P., 2004. An efficient algorithm for mining frequent sequences by a new strategy without support counting. *Proceedings of the 20th International Conference on Data Engineering*, Boston, USA, pp. 375-386.

Garriga, G.C., 2005. Summarizing sequential data with closed partial orders. *Proceedings of the SIAM International Conference on Data Mining*, Newport Beach, California, USA, pp. 380-391.

Gomez, L.I. and Vaisman, A.A., 2009. Efficient constraint evaluation in categorical sequential pattern mining for trajectory databases. *Proceedings of the 12th International Conference on Extending Database Technology*, Russia, pp. 541-552.

Joshi, M., Karypis, G. and Kumar, V., 2001. A universal formulation of sequential patterns. *Proceedings of the KDD 2001 Workshop on Temporal Data Mining*, San Francisco, USA.

Loekito, E., Bailey, J. and Pei, J., 2009. A binary decision diagram based approach for mining frequent subsequences. *Knowledge and Information Systems*, 24(2), pp. 235-268.

Lu, J., Wang, X.F., Adjei, O. and Hussain, F., 2004. Sequential patterns graph and its construction algorithm. *Chinese Journal of Computers*, 27(6), pp. 782-788.

Lu, J., Chen, W.R., Adjei, O. and Keech, M., 2008. Sequential patterns post-processing for structural relation patterns mining. *International Journal of Data Warehousing and Mining*, 4(3), pp. 71-89.

Lu, J., Chen, W.R. and Keech, M., 2010. Graph-based modelling of concurrent sequential patterns. *International Journal of Data Warehousing and Mining*, 6(2), pp. 41-58.

Mabroukeh, N.R. and Ezeife, C.I., 2010. A taxonomy of sequential pattern mining algorithms. *ACM Computing Surveys*, 43(1), pp. 1-41.

Pei, J., Han, J.W., Mortazavi-Asl, B. and Pinto, H., 2001. PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth. *Proceedings of 7th International Conference on Data Engineering*, Heidelberg, Germany, pp. 215-224.

Song, S., Hu, H. and Jin, S., 2005. HVSM: A new sequential pattern mining algorithm using bitmap representation. *Advanced Data Mining and Applications*, LNCS 3584, pp. 455-463.

van der Aalst, W.M.P., ter Hofstede, A.H.M., Kiepuszewski, B. and Barros, A.P., 2003. Workflow patterns. *Distributed and Parallel Databases*, 14(3), pp. 5-51.

Yang, Z., Wang, Y. and Kitsuregawa, M., 2007. LAPIN: Effective sequential pattern mining algorithms by last position induction for dense databases. *Advances in Databases: Concepts, Systems and Applications*, LNCS 4443, pp. 1020-1023.

Zaki, M.J., 2001. SPADE: An efficient algorithm for mining frequent sequences. *Machine Learning*, 42(1/2), pp. 31-60.