УДК 004.056

И. Ю. Трубицын, Я. А. Бекенёва
Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В. И. Ульянова (Ленина)

Формирование наборов данных при проведении секвенциального анализа событий

Представлена разработка и реализация адаптации алгоритма обнаружения и прогнозирования нарушений на основе частых наборов событий. Особенностью данной адаптации является формирование транзакций по времени событий, использование собственных скриптов, а также принятие минимальной границы для показателя достоверности, равной 80 %, что позволяет уменьшить количество ложных срабатываний. Обучающий набор данных, на котором модель создается и обучается, а затем тестируется и используется для предсказания значений в других наборах данных, представляет собой отметки о передвижении персонала по реальному предприятию через систему контроля доступа. На примере исследования данного набора показано, что благодаря предложенной адаптации можно выявлять нарушения и нетипичное поведение, проявляющиеся в невыполнении ассоциативных правил, которые создаются на основе частых наборов событий, полученных в результате работы алгоритма Frequent Pattern-Growth. Применимость предложенной адаптации подтверждена программным прототипом, разработанным на основе программной платформы для обработки данных RapidMiner и скриптов на языке программирования Groovy.

Интеллектуальный анализ данных, секвенциальный анализ событий, выявление нарушений, ассоциативные правила

Интеллектуальный анализ данных – процесс обнаружения полезных сведений из больших наборов данных. В ходе его проведения исследуются различные варианты событий, которые могут быть представлены в виде таблицы, строки которой представляют собой какой-либо вариант, а столбцы содержат характеризующие его параметры.

Добыча данных [1] (от *англ.* data mining) – процесс обнаружения в данных нетривиальных закономерностей, которые требуются для принятия решений во всевозможных сферах деятельности человека.

Технология Data Mining позволяет выявлять среди больших объемов данных взаимосвязи между отдельными событиями, основываясь на методах из теорий искусственного интеллекта, математической статистики, баз данных. Алгоритмы, используемые в данной технологии, требуют обильного количества вычислений, однако, учитывая рост производительности современных процессоров, а также объема оперативной памяти, можно провести качественный анализ миллионов записей за приемлемое время.

Обнаруживаемые знания должны быть [2]:

1. Ранее неизвестны – ресурсы, затраченные на открытие уже известных знаний, не окупаются.

2. Нетривиальны – знания не могут быть получены простыми способами и отражают неизвестные закономерности в данных.

3. Практически полезны – применяются с высокой точностью.

4. Доступны для понимания – представлены в комфортном для человека виде, а также логически объяснимы.

Классификация задач Data Mining. Методами Data Mining решаются следующие задачи [3]:

– классификации;

– регрессии;

– прогнозирования;

– кластеризации;

– определения взаимосвязей (поиск ассоциативных правил);

– анализ последовательностей;

– анализ отклонений.

Задачи интеллектуального анализа данных можно разделить по способу решения на обучение с учителем (от *англ.* supervised learning – контролируемое обучение) и без учителя (от *англ.* unsupervised learning – неконтролируемое обуче-

ние). При контролируемом обучении требуется некоторая обучающая выборка, на которой модель создается, обучается, тестируется и впоследствии применяется для предсказания значений в других наборах данных. При неконтролируемом обучении обучающего набора данных не требуется и главной задачей выступает обнаружение закономерностей в некотором существующем наборе данных [4].

К обучению с учителем относятся задачи прогнозирования, классификации и регрессии. Задача прогнозирования заключается в предсказании новых значений числовой последовательности на основании некоторых уже существующих. Задача классификации сводится к нахождению для каждого варианта некоторой категории или класса, к которому он принадлежит. Для этого требуется, чтобы множество классов было конечным, счетным и заранее известным. Задача регрессии подобна задаче классификации за исключением того, что в процессе ее решения проводится поиск шаблонов для определения числового значения, т. е. предсказываемый параметр представляет собой число из некоторого непрерывного диапазона [5].

К обучению без учителя относятся задачи определения взаимосвязей, анализа отклонений, анализа последовательностей, кластеризации. Задача определения взаимосвязей сводится к нахождению наиболее часто встречающихся наборов объектов среди множества подобных. Анализ отклонений сводится к поиску среди множества событий тех, которые существенно отличаются от нормы. Анализ последовательностей или секвенциальный анализ подразумевает обнаружение закономерностей в последовательностях событий. Задача кластеризации подразумевает разделение множества объектов на схожие по параметрам группы (кластеры), число которых может быть заранее неизвестно и определяется по совокупности параметров в процессе решения задачи [5]. Данные задачи также можно разделить по способу назначения на предсказательные и описательные. К предсказательным относятся задачи классификации, регрессии и определения взаимосвязей при условии, что полученные результаты могут быть применены для предсказания появления событий. К описательным – задачи кластеризации и определения взаимосвязей.

Предсказательные задачи решаются следующим образом: при помощи набора данных с известными результатами строится модель, которая затем применяется для предсказания результатов на основании некоторых новых наборов данных [4].

Цель описательных задач заключается в улучшении понимания человеком анализируемых данных.

Задача поиска ассоциативных правил. Главная цель определения взаимосвязей – нахождение между событиями частых ассоциаций (зависимостей), которые представляются в виде правил, впоследствии используемых для предсказания появления событий. Это один из наиболее используемых методов интеллектуального анализа данных.

Поиск ассоциативных правил представляется следующим образом. Объекты, которые формируют исследуемые наборы (от *англ.* itemsets), обозначаются в виде множества

$$I = \{i_1, i_2, \ldots, i_j, \ldots, i_n\},$$

где i_j – объекты, входящие в анализируемые наборы; n – общее количество объектов.

Транзакция – хранящийся в базе данных и подвергающийся анализу набор объектов из множества I. Описывается как подмножество множества I:

$$T = \{i_j \mid i_i \in I\}.$$

Транзакции, информация о которых доступна для анализа, обозначаются как множество

$$D = \{T_1,\ T_2,\ \ldots,\ T_r,\ \ldots,\ T_m\},$$

где m – количество транзакций, доступных для анализа.

Транзакции, в которые входит объект i_j, обозначаются как множество

$$D_{ij} = \{T_r \mid i_j \in T_r;\ j = 1...n;\ r = 1...m\} \subseteq D.$$

Некоторый произвольный набор объектов обозначается как

$$F = \{i_j \mid i_j \in I; j = 1...n\}.$$

Набор, состоящий из k объектов, называется *k-элементным набором.*

Транзакции, в которые входит набор F, обозначаются как множество

$$D_F = \{T_r \mid F \subseteq T_r;\ r = 1...m\} \subseteq D.$$

Отсюда, поддержкой (от *англ.* support) набора F является отношение количества транзакций, в которые входит набор F, к общему количеству транзакций:

$$\mathrm{Supp}(F) = \frac{|D_F|}{|D|}.$$

Если значение поддержки набора больше минимального явно заданного значения поддержки, то такой набор считается *частым* (от *англ.* large itemset): $\mathrm{Sup}(F) > \mathrm{Sup}_{\min}$.

Таким образом, поиск ассоциативных правил сводится к нахождению множества всех частых наборов [6]:

$$L = \{F \mid \mathrm{Sup}(F) > \mathrm{Sup}_{\min}\}.$$

Представление результатов при решении задачи поиска ассоциативных правил. При решении задачи поиска ассоциативных правил выделяют следующие этапы:

– нахождение всех частых наборов объектов;

– генерация ассоциативных правил на основе найденных частых наборов объектов.

Ассоциативное правило выглядит следующим образом:

$$\text{ЕСЛИ}(\text{условие})\,\text{ТО}(\text{результат}),$$

где условие – набор объектов из множества I, с которыми ассоциированы объекты, включенные в результат данного правила.

Таким образом, условие и результат есть объекты множества I: ЕСЛИ X ТО Y, где $X \in I,\ Y \in I,\ X \cup Y = \varphi$.

Ассоциативное правило может быть представлено как импликация над множеством $X \Rightarrow Y$, где $X \in I,\ Y \in I,\ X \cup Y = \varphi$, т. е. закономерностью вида: «Если в транзакции встречается набор X, значит, в этой транзакции должен появиться набор Y».

Существуют специальные величины для оценки полезности ассоциативных правил: поддержка, достоверность, улучшение. Данные величины используются при генерации правил: им задаются минимальные значения; те правила, которые не соответствуют условиям, не используются в решении задачи [6]. Если объекты имеют дополнительные атрибуты, которые могут влиять на состав объектов в транзакциях, они тоже учитываются.

Поддержка (от *англ.* support) отражает процент транзакций, поддерживающих данное правило:

$$\mathrm{Sup}_{X \Rightarrow Y} = \mathrm{Sup}_F = \frac{|D_{F=X \cup Y}|}{|D|}$$

Поскольку построение правил производится на основании набора, то правило $X \Rightarrow Y$ имеет поддержку, которая равна поддержке набора F, состоящего из X и Y.

Достоверность (от *англ.* confidence) показывает вероятность существования в транзакции набора Y на основе существования в ней набора X. Достоверность правила $X \Rightarrow Y$ представляется в виде отношения числа транзакций, содержащих наборы X и Y, к числу транзакций, содержащих только набор X:

$$\mathrm{Conf}_{X \Rightarrow Y} = \frac{|D_{F=X \cup Y}|}{|D_X|} = \frac{\mathrm{Sup}_{X \cup Y}}{\mathrm{Sup}_X}.$$

Улучшение (от *англ.* improvement) показывает, оправданно ли использование правила. Представляется в виде отношения числа транзакций, содержащих наборы X и Y, к произведению количества транзакций, содержащих только набор X, и количества транзакций, содержащих только набор Y:

$$\mathrm{impr}_{X \Rightarrow Y} = \frac{|D_{F=X \cup Y}|}{|D_X||D_Y|} = \frac{\mathrm{Sup}_{X \cup Y}}{\mathrm{Sup}_X * \mathrm{Sup}_Y}.$$

Правила, построенные на основании одного и того же набора, имеют одинаковое значение поддержки. Если значение поддержки велико, то в результате будут найдены очевидные правила, если мало – будут найдены необоснованные правила.

Чем больше значение достоверности правила, тем лучше. Правила, построенные на основании одинакового набора, имеют разную достоверность.

Однако если значение достоверности ниже, чем значение (в процентах) безусловного наличия набора Y, тогда существует вероятность случайного отгадывания наличия в транзакции набора Y:

$$\mathrm{Conf}_{X \Rightarrow Y} = \frac{\mathrm{Sup}_{X \cup Y}}{\mathrm{Sup}_X} < \mathrm{Sup}_Y.$$

Если значение улучшения больше единицы, то использование правила оправданно, иначе – нет. Если использование правила не оправданно, то можно использовать отрицающее правило.

Поскольку ассоциативные правила строятся на основе частых наборов, их не всегда можно применить, к тому же их количество может быть велико вследствие того, что правила, построенные на основании набора F, представляют собой все возможные комбинации объектов, входящих в данный набор. Однако главное достоинство ассоциативных правил – их наглядность и интерпретируемость [6].

Секвенциальный анализ. Задача поиска частых наборов не учитывает время как атрибут транзакции, однако зачастую существует необходимость анализа последовательности протекающих собы-

тий. Секвенциальный анализ позволяет предсказывать с определенной долей вероятности будущие события, так как отличается от задачи поиска ассоциативных правил установлением отношения порядка между исследуемыми наборами [7].

Последовательность протекающих событий представляет собой упорядоченное множество объектов. Если на множестве задано отношение порядка, то последовательность можно описать следующим образом:

$$S = \{\ldots,\ i_p,\ \ldots,\ i_q\},\ p < q\ .$$

Последовательности бывают с циклами и без. В первом случае один и тот же объект может входить в последовательность несколько раз, но на разных позициях:

$$S = \{\ldots,\ i_p,\ \ldots,\ i_q\},\ p < q,\ i_p = i_q\ .$$

Последовательность S может входить в транзакцию T, если $S \subseteq T$ и объекты из S содержатся во множестве T с сохранением отношения порядка. Однако при этом во множестве T между объектами последовательности S могут находиться другие объекты [6].

Поддержка последовательности S определяется как отношение количества транзакций, которые содержат последовательность S, к общему количеству транзакций.

Последовательность считается частой, если ее поддержка превышает минимальную заданную поддержку:

$$\mathrm{Sup}(S) > \mathrm{Sup}_{\min}\ .$$

Таким образом, задача секвенциального анализа сводится к поиску всех частых последовательностей:

$$L = \{S \mid \mathrm{Sup}(S) > \mathrm{Sup}_{\min}\}\ .$$

Алгоритм Apriori. Смысл данного алгоритма заключается в использовании свойства поддержки: поддержка любого набора объекта не превышает минимальную поддержку любого из его подмножеств: $\mathrm{Sup}_F \le \mathrm{Sup}_E$ при $E \subset F$.

Алгоритм находит часто встречающиеся наборы в несколько этапов, каждый из которых состоит из последовательности шагов: формирования кандидатов и их подсчета [6].

Недостатком данного алгоритма является процесс генерации кандидатов, который требует больших вычислительных и временных затрат. К тому же, он требует многократного сканирования базы данных и не учитывает временную составляющую в наборах данных. Поэтому у данного алгоритма существует множество модификаций.

Apriori – это один из первых алгоритмов, рассчитанных на решение задачи поиска ассоциативных правил [2].

Алгоритм Frequent Pattern-Growth (FPG). Один из наиболее эффективных алгоритмов поиска ассоциативных правил [8], который позволяет не использовать затратную процедуру генерации кандидатов.

Основным достоинством FPG-метода является сжатие базы данных транзакций за счет использования дерева популярных предметных наборов (Frequent-Pattern Tree). К тому же в нем гарантируется полное извлечение частых наборов.

В FP-дереве каждый элемент представляется в виде узла с индексом, указывающим на количество его повторений. Это – наиболее выгодный вариант представления базы данных транзакций, так как в ней каждый элемент может неоднократно повторяться [9].

С увеличением количества транзакций в базе данных временные затраты на поиск частых наборов для алгоритма Apriori растут на несколько порядков быстрее, чем для алгоритма FPG.

Подготовка данных для секвенциального анализа событий. Для успешного проведения секвенциального анализа событий удобно воспользоваться инструментом для обработки данных. В данной статье в качестве такого программного обеспечения выступает платформа RapidMiner.

RapidMiner [9] – инструмент с открытым исходным кодом (open-source) для обработки данных, идея которого заключается в том, что аналитик может не прибегать к программированию при выполнении работы.

Исходными данными для анализа служит набор событий, которые представляют собой отметки о передвижении персонала по предприятию через систему контроля доступа. Данные представлены в формате «.csv» (CSV – Comma-Separated Values) – это текстовый формат, который предназначен для представления табличных данных.

Для точного нахождения частых наборов событий необходимо подходящее формирование транзакций, которое может быть реализовано несколькими способами:

id	department	zone	wd	time_shift	duration	transId
junger002	Information Technology	1-1	5	1921	50000	1
junger002	Information Technology	1-4	5	1971	42000	1
junger002	Information Technology	2-4	5	2013	2000	1

Рис. 1

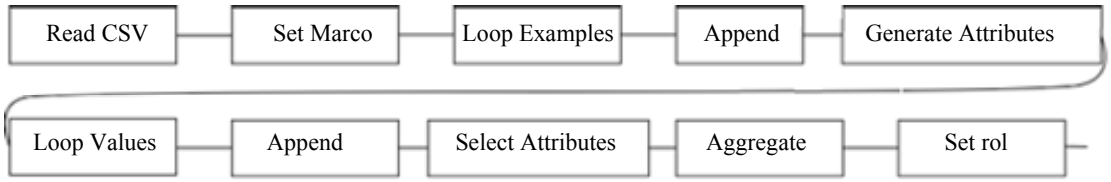

Рис. 2

Рис. 3

– по маркерам;

– по времени;

– скользящим окном.

В данной работе используется способ, основанный на формировании транзакций по времени. Каждая строка в исходных данных представляет собой событие – отметку о передвижении сотрудника. События, относящиеся к одному и тому же сотруднику и происходящие в один день, считаются одной транзакцией. Такие события имеют одинаковое значение идентификатора транзакции – поле transId (рис. 1).

Для каждого события добавляется поле action, которое представляет собой конкатенацию* полей, входящих в транзакцию [10].

Для событий, имеющих одинаковое значение идентификатора транзакции, производится конкатенация полей action – таким образом формируются транзакции (рис. 2)

В данной статье используется обучающий набор данных, на котором модель данных создается, обучается, тестируется и впоследствии используется для предсказания значений в других наборах данных.

Из обучающего набора функциональный блок Read CSV считывает данные. Функциональный блок Set Macro представляет собой инициализацию глобальной переменной transID – значения идентификаторов транзакций.

Блок Loop Examples представляет собой цикл (рис. 3). В данном цикле к каждой строке из входных данных добавляется поле со значением идентификатора транзакции, а также создаются поля begin и dur для конвертации значений полей time_shift и duration из миллисекунд в минуты. На каждой итерации данный блок возвращает одиночную строку.

Функциональный блок Append выполняет объединение полученных одиночных строк в один набор данных.

Затем функциональный блок Generate Attributes производит последующую генерацию значений поля action. Данное поле представляет собой конкатенацию значений полей zone, wd, begin и dur, разделенных знаком «_».

Блок Loop Values представлен на рис. 4. В данной группе функциональных блоков события, имеющие одинаковое значение поля идентификатора транзакции, сортируются в порядке возрастания значений поля time_shift. Данный блок возвращает наборы событий, относящихся к одной транзакции.

Рис. 4

Функциональный блок Append выполняет объединение полученных наборов данных в один набор.

Функциональный блок Select Attributes осуществляет выборку необходимых полей, используемых для конечного формирования транзакций – transId, action.

Блок Aggregate выполняет конкатенацию полей action, формируя конечные транзакции.

* Конкатенация – объединение нескольких объектов.

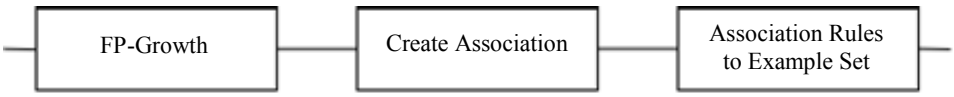

Рис. 5

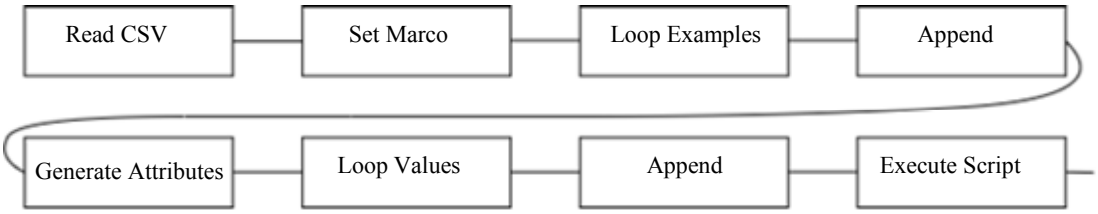

Рис. 6

transId ↓	concat(action)
99	3-1 _3 _0.3 _0.6|3-4 _3 _0.3 _0.5|2-4 _3 _0.3 _0.5|0-0 _3 _0.3 _883.1|1-4 _3 _0.3 _0|1-1 _3 _0.3 _0.1|1-1 _3 _0.5 _1.1|1-4 _3 _0.5 _0.5|2-4 _3 _0.5 _0.5|3-4 _3 _
98	1-1 _5 _0.5 _1.6|1-4 _5 _0.5 _0.7|2-4 _5 _0.5 _0|2-1 _5 _0.5 _0.3|2-7 _5 _0.5 _81.9|2-1 _5 _0.5 _8|2-7 _5 _0.6 _74.6|2-6 _5 _0.6 _0
97	1-1 _4 _0 _1.5|1-4 _4 _0 _0.7|2-4 _4 _0 _0|2-1 _4 _0 _0.3|2-7 _4 _0 _74.5|2-6 _4 _0.1 _55.2|2-7 _4 _0.2 _71.2|2-1 _4 _0.2 _12|2-7 _4 _0.3 _38|2-1 _4 _0.3 _1|2-4

Рис. 7

Size	Support ↑	Item 1	Item 2
1	0.051	1-1 _4 _0.5 _1.6	
1	0.051	1-1 _5 _0.5 _1.6	
1	0.051	2-4 _2 _0.7 _0.7	
1	0.051	2-4 _3 _0.4 _0	
2	0.051	1-4 _5 _0.3 _0	2-1 _5 _0.5 _0.3
2	0.051	1-4 _5 _0.5 _0.7	1-1 _5 _0.5 _1.6
2	0.051	1-4 _2 _0.5 _0.7	1-1 _2 _0.3 _0.1
2	0.051	1-4 _2 _0.5 _0.7	2-1 _2 _0.1 _0.3

Рис. 8

Блок Set Role назначает полю идентификатора транзакции роль «id», что необходимо для корректной работы функционального блока, отвечающего за алгоритм FPG.

Таким образом формируются транзакции.

Поиск нарушений. Реализация группы функциональных блоков, отвечающих за анализ и поиск ассоциативных правил, приведена на рис. 5.

Функциональный блок FP-Growth представляет собой реализацию алгоритма FPG, возвращаемое значение которого – частые наборы событий.

На основе наборов, полученных в результате работы функционального блока FP-Growth, блок Create Association создает ассоциативные правила. В данном блоке задано минимальное значение достоверности 80 %, так как при более низком значении возможны частые ложные срабатывания, а использование более высокого значения необоснованно.

Функциональный блок Association Rules to Example Set приводит к формату, удобному для дальнейшей работы.

На основе полученных ассоциативных правил проводится поиск нарушений (рис. 6).

Функциональный блок Execute Script позволяет выполнять собственный скрипт на языках программирования Java и Groovy. Данный блок выполняет поиск нарушений найденных правил в исходных данных.

Тестирование разработанной адаптации алгоритма. В рамках данной работы тестирование проводилось на основе реальных данных предприятия. В качестве обучающего набора использовались первые 10 890 записей исходного файла, а в качестве проверяемых событий выступали оставшиеся 19 190 записей.

Пример формирования транзакций представлен на рис. 7, а пример формирования частых наборов – на рис. 8.

Пример формирования ассоциативных правил на основе обучающего набора приведен на рис. 9, где Premises – предпосылка, Conclusion – следствие, Support – поддержка, Confidence – достоверность, Lift – улучшение.

Premises	Conclusion	Support ↓	Confidence	Lift
1-1 _4_0.3_0.1	1-4 _4_0.3_0	0.157	0.970	6.182
1-4 _4_0.3_0	1-1 _4_0.3_0.1	0.157	1	6.182
1-1 _3_0.3_0.1	1-4 _3_0.3_0	0.152	0.969	6.375
1-4 _3_0.3_0	1-1 _3_0.3_0.1	0.152	1	6.375
1-1 _5_0.3_0.1	1-4 _5_0.3_0	0.147	0.968	6.581
1-1 _6_0.3_0.1	1-4 _6_0.3_0	0.147	0.968	6.581
1-4 _5_0.3_0	1-1 _5_0.3_0.1	0.147	1	6.581
1-4 _6_0.3_0	1-1 _6_0.3_0.1	0.147	1	6.581
2-4 _5_0.3_0.7	1-1 _5_0.3_0.1	0.110	1	6.581

Рис. 9

lazada001	Engineering	2-1	6	0.7	23.6	2-1 _6_0.7_23.6	
lazada001	Engineering	2-4	6	0.7	0.7	2-4 _6_0.7_0.7	Правило2-4 _6_0.7_0.7 -> 1-4 _6_0.7_0
lazada001	Engineering	1-4	6	0.7	0	1-4 _6_0.7_0	

Рис. 10

pyoung002	Facilities	0-0	6	0.3	3766.4	0-0 _6_0.3_3766.4	
pyoung002	Facilities	1-4	6	0.3	0	1-4 _6_0.3_0	Возможное нарушение: 1-4 _6_0.3_0 -> 1-1 _6_0.3_0.1 ...
pyoung002	Facilities	1-1	6	0.3	0.4	1-1 _6_0.3_0.4	

а

pyoung002	Facilities	1-1	3	0.1	0.9	1-1 _3_0.1_0.9	
pyoung002	Facilities	1-4	3	0.1	0.7	1-4 _3_0.1_0.7	Возможное нарушение: 1-4 _3_0.1_0.7 -> 1-1 _3_0.3_0.1
pyoung002	Facilities	2-4	3	0.1	0.5	2-4 _3_0.1_0.5	

б

Рис. 11

Полученные ассоциативные правила применяются к проверяемым событиям. Возможно несколько случаев:

– для события может не оказаться никакого ассоциативного правила;

– ассоциативное правило может быть найдено и выполнено для данного события;

– ассоциативное правило может быть найдено и не выполнено для данного события, что может расцениваться как нарушение с какой-либо долей вероятности.

Пример выполнения ассоциативного правила для события приведен на рис. 10. Данное правило имеет вид «2-4_6_0.7_0.7→1-4_6_0.7_0», что означает, что после события 2-4_6_0.7_0.7 должно произойти событие 1-4_6_0.7_0.

Примером невыполнения ассоциативных правил может быть нетипичное время (рис. 11, *а*) и нетипичный переход между зонами (рис. 11, *б*).

Для решения задачи обнаружения и прогнозирования нарушений был выбран вариант адаптации существующих алгоритмов поиска ассоциативных правил. В отличие от разработки нового алгоритма данный подход менее трудоемкий и основывается на использовании проверенных решений и методов.

Для проведения секвенциального анализа событий был задействован инструмент обработки данных – платформа RapidMiner, в которой с помощью функциональных блоков была реализована адаптация алгоритма FPG. Этот алгоритм поиска ассоциативных правил наиболее эффективен, так как он гарантирует полное извлечение данных вследствие сжатия базы данных транзакций до FP-дерева.

В качестве основных аспектов для успешного проведения анализа были выделены: формирование транзакций по времени событий, совершенных сотрудником; минимальное значение достоверности, принятое равным 80 % для снижения количества ложных срабатываний; для поиска нарушений использовались собственные скрипты на языке программирования Groovy.

Авторы рассмотрели полученный результат и определили дальнейшее развитие разработанного алгоритма адаптации, которое может быть направлено на его распараллеливание, что обеспечит более эффективное распределение ресурсов.

СПИСОК ЛИТЕРАТУРЫ

1. Data Mining – добыча данных. URL: https://basegroup.ru/community/articles/data-mining (дата обращения 25.05.2019).

2. Барсегян А. А., Куприянов М. С., Холод И. И. Анализ данных и процессов: учеб. пособие. 3-е изд., перераб. и доп. СПб.: БХВ-Петербург, 2009.

3. Барсегян А. А. Технологии анализа данных. СПб.: БХВ-Петербург, 2007.

4. Тематические основы бизнес-аналитики. URL: http://samzan.ru/228414 (дата обращения 25.05.2019).

5. Дядичев В. В., Ромашка Е. В., Голуб Т. В. Задачи и методы интеллектуального анализа данных // Геополитика и экогеодинамика регионов. 2015. Т. 1, № 3. С. 23–29.

6. Матвейкин В. Г., Дмитриевский Б. С., Ляпин Н. Р. Информационные системы интеллектуального анализа. М.: Машиностроение, 2008.

7. Секвенциальный анализ. Методы и средства анализа данных. URL: http://bourabai.ru/tpoi/analysis5.htm (дата обращения 25.05.2019).

8. FPG – альтернативный алгоритм поиска ассоциативных правил | BaseGroup Labs: URL: https://basegroup.ru/community/articles/fpg (дата обращения 25.05.2019).

9. RapidMiner – платформа для анализа больших данных | Бизнес-архитектура. URL: https://businessarchitecture.ru/rapidminer/ (дата обращения 25.05.2019).

10. Конкатенация строк в Java. URL: https://vertex-academy.com/tutorials/ru/konkatenaciya-strok/ (дата обращения 25.05.2019).

I. Yu. Trubitsyn, Ya. A. Bekeneva
Saint Petersburg Electrotechnical University

FORMATION OF DATABASES DURING SEQUENTIAL EVENT ANALYSIS

The development and implementation of adaptation of the algorithm for detecting and predicting violations based on frequent sets of events is presented. The peculiarity of this adaptation is the formation of transactions by the time of events, the use of their own scripts, as well as the adoption of a minimum border for a confidence indicator of 80%, which reduces the number of false positives. The training data set, on which the model is created and trained, and then tested and used to predict values in other data sets, is a mark on the movement of personnel in a real enterprise through an access control system (ACS). An example of a study of this set shows that, thanks to the proposed adaptation, it is possible to detect violations and atypical behavior, manifested in the failure to comply with the associative rules, which are created on the basis of frequent sets of events obtained as a result of the work of the Frequent Pattern-Growth algorithm. The applicability of the proposed adaptation is confirmed by a software prototype developed on the basis of a software platform for processing RapidMiner data and scripts in the Groovy programming language.

Data Mining, sequential event analysis, violation detection, associative rules