УДК 681.003.66

А.Л. Стокипный

Харьковский университет Воздушных Сил им. И. Кожедуба, Харьков

СПОСОБ ЭФФЕКТИВНОГО ПРЕДСТАВЛЕНИЯ ИССЛЕДУЕМОГО НАБОРА ДАННЫХ В МЕТОДАХ ПОИСКА АССОЦИАТИВНЫХ ПРАВИЛ

В статье дано описание способа представления исследуемого набора данных в методах поиска замкнутых часто встречающихся наборов элементов, основанных на использовании FP-tree подобных структур данных. По результатам экспериментов применение предложенного способа во всех тестовых наборах позволяет достигнуть сокращения объема памяти, необходимого для размещения исследуемого набора данных, в среднем от 2 до 12 раз. Разница во времени построения структуры в оперативной памяти стандартным способом и предложенным относительно общего времени поиска ассоциативных правил является несущественной.

Ключевые слова: *Data Mining, ассоциативное правило, часто встречающийся набор элементов, FP-tree.*

Введение

На сегодняшний день ввиду постоянного роста объемов информации, накопленной в хранилищах данных и отражающей результаты деятельности человека в различных областях, широкое развитие получила концепция интеллектуального анализа данных (Data Mining).

Суть Data Mining состоит в нахождении новых, действительных и потенциально полезных знаний в больших объемах первоначальных (сырых) данных [1].

Одним из ключевых направлений современных исследований, связанных с Data Mining, является разработка методов поиска ассоциативных правил, посредством которых выявляются и представляются скрытые связи между объектами некоторой предметной области [2].

Уже сейчас имеют место факты успешного применения методов поиска ассоциативных правил в таких областях как здравоохранение [3], финансы [4], банковское дело [5], телекоммуникации [6], биология [7] и множестве других [8, 9, 10].

Пусть $I = i...i_{N^I}$ – множество элементов, которые формируют отдельную запись набора данных $T = t...t_{N^T}$, где N^I – общее количество элементов, N^T – общее количество записей, $t \subseteq I$.

Определение 1. Ассоциативным правилом называется импликация вида $X \Rightarrow Y$, где X, Y – непересекающиеся наборы элементов множества I ($X \subset I$, $Y \subset I$, $X \cap Y = 0$) [11, 12].

Каждое ассоциативное правило оценивается с помощью двух показателей: поддержки и достоверности.

Определение 2. Запись t содержит набор элементов (НЭ) X, если $X \subseteq t$. Количество записей, которые содержат НЭ X, называется поддержкой НЭ X и обозначается $\sup X$.

В терминах теории вероятности поддержка $\sup X$ представляет собой вероятность наступления события $X \subseteq t$.

Определение 3. Поддержкой ассоциативного правила $X \Rightarrow Y$, обозначается $\sup X \Rightarrow Y$, называется вероятность наступления события $X \cup Y \subseteq t$, $\sup X \Rightarrow Y = \sup X \cup Y$.

Определение 4. Достоверностью ассоциативного правила $X \Rightarrow Y$, обозначается $\mathrm{conf}\ X \Rightarrow Y$, называется условная вероятность наступления события $Y \subseteq t$ при условии, что наступило событие $X \subseteq t$, $\mathrm{conf}\ X \Rightarrow Y = \sup X \cup Y / \sup X$.

Пусть заданы пороговые значения поддержки и достоверности, соответственно *min_sup* и *min_conf*, тогда задача поиска ассоциативных правил в наборе данных T сводится к решению двух подзадач [11, 12]:

1. Поиск всех НЭ $X \in I$, для которых $\sup X \geq min\_sup$.

2. Для каждого найденного на первом этапе НЭ X генерация ассоциативных правил вида $X' \Rightarrow X \setminus X'$, причем $X' \subset X$ и $\mathrm{conf}\ X' \Rightarrow X \setminus X' \geq min\_conf$.

Определение 5. Для заданного порогового значения поддержки *min_sup*, НЭ X называется часто встречающимся НЭ (ЧВНЭ), если $\sup X \geq min\_sup$.

Исходя из определения 5, первый этап поиска ассоциативных правил сводится к поиску ЧВНЭ. На сегодняшний день предложен ряд методов [13 – 17], которые для заданного значения поддержки *min_sup* в исследуемом наборе данных находят все ЧВНЭ.

© А.Л. Стокипный

Рассмотрим пример [19], в котором набор данных содержит только 2 записи, $i_1, i_2, ..., i_{100}$, $i_1, i_2, ..., i_{50}$, порог поддержки *min_sup=1* и порог достоверности *min_conf=0,5*. Обычные методы поиска ЧВНЭ сгенерируют $2^{100} - 1 \approx 10^{30}$ НЭ, на основе которых затем будет сформирован достаточно большой объем ассоциативных правил. В работе [11] предложен альтернативный подход: вместо поиска полного набора ЧВНЭ и соответствующих ассоциативных правил, необходимо найти только замкнутые ЧВНЭ и сгенерировать на их основе множество ассоциативных правил.

Определение 6. ЧВНЭ X называется замкнутым, если не существует НЭ X', такого, что $X' \supset X$ и $\sup X' = \sup X$.

Возвращаясь к вышеописанному примеру, методы поиска замкнутых ЧВНЭ в качестве результаты возвратят только 2 ЧВНЭ: $i_1, i_2, ..., i_{100}$ и $i_1, i_2, ..., i_{50}$, на основе которых будет сгенерировано одно ассоциативное правило $i_1, i_2, ..., i_{50} \Rightarrow i_1, i_2, ..., i_{100}$. Оставшиеся ЧВНЭ и ассоциативные правила могут быть затем легко получены из замкнутых простым перебором их элементов.

Анализ методов поиска замкнутых часто встречающихся наборов элементов

Анализ публикаций показал, что существующие на сегодняшний день методы поиска замкнутых ЧВНЭ [18] можно разделить на две группы:

- методы с последовательной генерации кандидатов;
- методы, использующие парадигму «разделяй и властвуй».

Определение 7. Набор элементов

$$X\ k \subseteq I, k = \overline{1...N^I}$$

называется k-уровневым, если он содержит k неповторяющихся элементов множества I.

В основе методов первой группы лежит так называемый Apriori-подобный принцип поиска. Данный подход, который некоторые авторы называют поиском в ширину [20], состоит в том, что потенциальные l-уровневые ЧВНЭ генерируются на основе ЧВНЭ уровня $l-1$. Для определения поддержки сгенерированных потенциальных ЧВНЭ на каждом уровне необходим просмотр всех записей из T. То есть, все записи набора данных просматриваются L раз, где L – количество элементов в ЧВНЭ наивысшего уровня, который присутствует в результирующем наборе. Основным недостатком поиска в ширину является необходимость многократного обращения ко всем записям набора данных и промежуточного хранения всех потенциальных кандидатов, что в случае большой длины ЧВНЭ и/или низких значений поддержки ведет к существенному снижению эффективности [15, 20].

Представителями данного подхода являются методы Close, A-close [11, 12].

Методы второй группы в общем случае используют алгоритм BackTracking, представленный ниже. Различия состоят лишь в способах представления исследуемого набора данных и стратегиях отсечения рекурсивных вызовов, которые заведомо не дадут новых замкнутых ЧВНЭ. В публикациях, относящихся к проблеме поиска ЧВНЭ, данный подход называют поиском в глубину [20].

Вход: X – ЧВНЭ

АЛГОРИТМ BackTracking (X)

1. **If** $\neg\exists c \in C,\ X \subset c,\ \sup c = \sup X$ **then begin**
2. $C = C \cup X$
3. **print** X
4. **end**
5. **For each** $i \in I / X$ **do**
6. **If** $\sup X \cup i > min\_sup$ **then**
7. **call** BackTracking ($X \cup i$)

Методы LCM [24] и DCI_closed [21] не сохраняют множество найденных замкнутых ЧВНЭ для дальнейшего его использования с целью проверки существования для текущего набора-кандидата надмножества с идентичным значением поддержки (строка №1 алгоритма BackTracking). Указанная особенность сокращает время поиска замкнутых ЧВНЭ и объем используемой оперативной памяти. Вместе с тем, на втором этапе для эффективной генерации ассоциативных правил нам необходимо организовать поиск во множестве замкнутых ЧВНЭ, которые получены на первом этапе в результате работы метода поиска замкнутых ЧВНЭ. Следовательно, выиграв в случае применения методов LCM или DCI_closed на первом этапе, мы будем вынуждены затратить на втором этапе определенные ресурсы для организации полученных замкнутых ЧВНЭ в структуру, которая бы позволяла выполнять эффективный поиск.

Горизонтальное представление исследуемого набора данных (рис. 1, а) характеризуется тем, что при его использовании каждая запись содержит идентификатор и множество элементов, которые входят в ее состав. Следует заметить, что к горизонтальным также относятся представления исследуемого набора данных в виде лучевых (trie) структур [20, 22, 27].

	i_1	i_2	i_3	i_4
1	0	1	1	0
2	0	1	1	1
3	1	1	0	1
4	1	0	0	1

а

	i_1	i_2	i_3	i_4
1	0	1	1	0
2	0	1	1	1
3	1	1	0	1
4	1	0	0	1

б

Рис. 1. Горизонтальный (а) и вертикальный (б) способы представления набора данных

При вертикальном представлении для каждого элемента $i \in I$ формируется покрытие, то есть список идентификаторов записей, в состав которых входит элемент $i \in I$. Покрытие может быть представлено как в виде списка идентификаторов [25, 23, 24], так и в битовом виде [21].

Эффективность работы метода поиска замкнутых часто-встречающихся множеств современные исследователи оценивают по двум основным показателям [30, 17]:

1. количество времени, затраченного на поиск замкнутых ЧВНЭ;

2. объем оперативной памяти, необходимый для работы метода.

В рамках 3-й и 4-й Международных Конференций по Data Mining [28, 29] были проведены исследования методов поиска ЧВНЭ различных типов. Результаты [30] показали, что среди методов поиска замкнутых ЧВНЭ в 7 из 14 тестовых наборов данных при высоком значении поддержки наилучший результат как по критерию использования оперативной памяти, так и по критерию времени работы показала реализация [31] метода FPclose [22], основанного на использовании специальной структуры данных FP-tree [15]. В других наборах более эффективными были реализации методов LCM [23, 24] и AFOPT [26]. При низком значении поддержки наилучшие результаты показали FPClose и LCM. Таким образом, двумя наиболее универсальными с точки зрения использования с различными тестовыми наборами данных при различных значениях поддержки, являются методы FPclose и LCM.

Как было указано ранее, в статье для генерации ассоциативных правил нам необходимо располагать множеством ранее выявленных замкнутых ЧВНЭ и иметь возможность эффективно выполнять в нем поиск. Именно такие возможности, в отличие от метода LCM, предоставляет метод FPclose и используемая в нем структура данных FP-tree.

Узким местом всех методов, в том числе и FPclose, использующих поиск в глубину, является необходимость размещения исследуемого набора данных в оперативной памяти [17]. Только в случае, если весь набор данных может быть размещен в оперативной памяти, методы второй группы показывают заявленные высокие результаты. То есть актуальной является проблема разработки структур данных, позволяющих эффективно размещать в оперативной памяти исследуемый набор данных для дальнейшего поиска в нем ЧВНЭ.

Целью статьи является разработка эффективного способа представления исследуемого набора данных в методах поиска замкнутых ЧВНЭ, основанных на использовании FP-tree подобных структур данных.

Далее в статье мы предложим FP-tree подобную структуру данных, которая при несущественном увеличении времени работы метода FPclose, позволяет существенно сократить объем оперативной памяти, используемой в ходе поиска замкнутых ЧВНЭ. Мы не будем останавливаться на особенностях реализации самого метода FPclose, которые детально описаны в работе [15], однако подробно исследуем саму структуру FP-tree, алгоритм ее построения и предлагаемые нами модификации.

Разработка способа представления исследуемого набора данных в методах поиска замкнутых ЧВНЭ

Табл. 1 представляет собой набор записей, который мы будем использовать в качестве примера. Пороговое значение поддержки в нашем примере равно *min_sup*=2.

Таблица 1

Тестовый набор записей

TID	Записи набора данных	Записи с отсортированными элементами
1	b, s, m, o, g, a, h, q	g, h, a, m, s
2	w, a, s, v, h, g	g, h, a, v, s
3	k, v, g, p, d, y	g, v, p
4	z, x, g, v, n, m	g, v
5	j, m, a, h, u, e, t	h, a, m

Третий столбец содержит записи исследуемого набора, из которых удалены элементы с поддержкой меньше заданной *min_sup*=2. Также элементы в записях третьего столбца отсортированы по убыванию значений поддержки, как требует алгоритм построения FP-tree [15]. В общем виде структура данных FP-tree может быть определена следующим образом:

- FP-tree содержит один узел, называемый «корнем» («root»), множество префиксных поддеревьев, являющихся потомками «корня», и заголовочную таблицу частых элементов.
- Каждый узел в префиксном дереве состоит из пяти полей:

1) идентификатор элемента, содержит числовой индекс элемента $i \in I$ или символьное обозначение;

2) значение поддержки, соответствующее количеству записей, которые содержат элементы, входящих в путь от «корня» к данному узлу;

3) список узлов-потомков;

4) указатель на родительский узел;

5) указатель на узел с таким же идентификатором элемента или значение «null», если такого узла не существует.

- Каждая запись в заголовочной таблице частых элементов состоит из двух полей:

1) идентификатор элемента;

2) указатель на узел, идентификатор которого равен идентификатору, указанному в поле №1.

Для тестового набора записей (табл. 1) соответствующая структура FP-tree представлена на рис. 2.

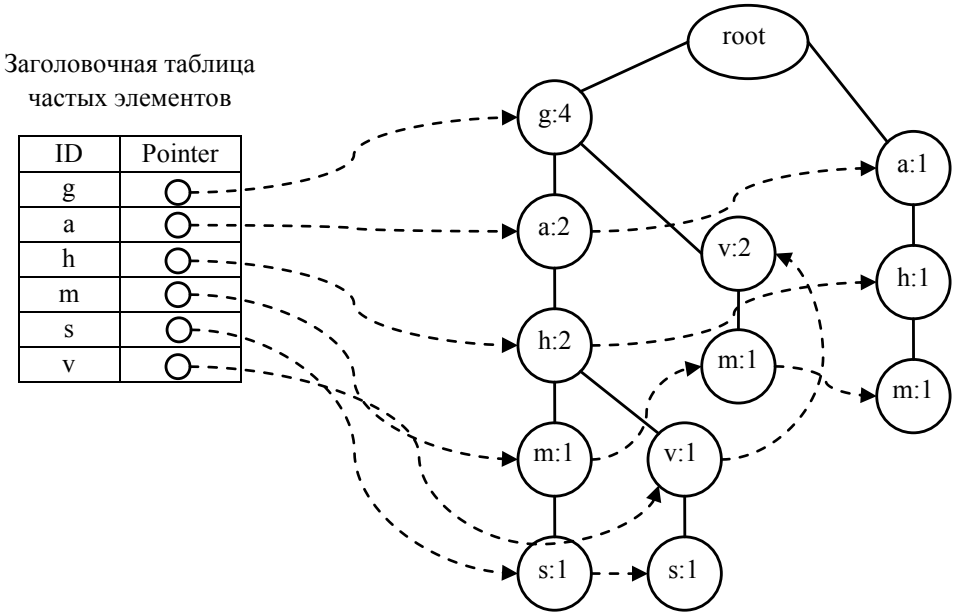


Рис. 2. FP-tree для тестового набора (табл. 1)

В методе поиска ЧВНЭ PatriciaMine [32] авторы положили в основу не обычную префиксную древовидную структуру, а ее модификацию – Patricia-tree. Отличие Patricia tree состоит в том, что узлы с одинаковым значением поддержки сливаются в один узел. Структура узла, таким образом, меняется, и поле №1, которое ранее содержало идентификатор элемента, теперь содержит связанный список идентификаторов элементов с одинаковым значением поддержки (рис. 3).

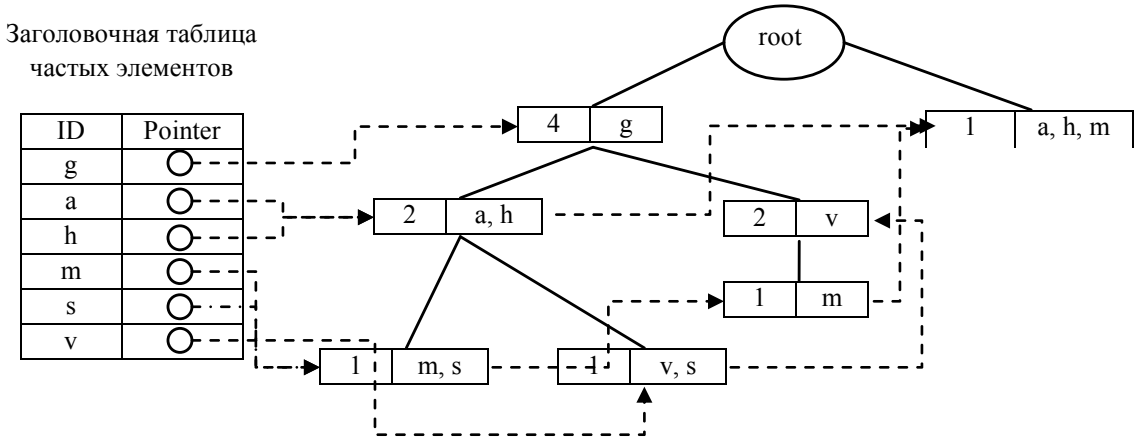


Рис. 3. Patricia-tree для тестового набора (табл. 1)

Программная реализация FP-tree в 32-разрядной среде использует 24-байтовую структуру Fnode для представления узла префиксного дерева.

СТРУКТУРА Fnode {

32-разрядное слово: идентификатор элемента;

32-разрядный указатель на структуру Fnode: родитель узла;

32-разрядный указатель на структуру Fnode: крайний левый потомок;

32-разрядный указатель на структуру Fnode: следующий узел с таким же родителем;

32-разрядный указатель на структуру Fnode:

следующий узел с таким же идентификатором элемента;

32-разрядный слово count: значение поддержки

}

В случае использование Patricia-tree узел префиксного дерева может быть реализован в виде структуры Pnode, размер которой равен 28 байтам.

СТРУКТУРА Pnode {

32-разрядное слово: размер массива идентификаторов элементов;

32-разрядный указатель на массив 32-разрядных слов: идентификаторы элементов с одинаковым значением поддержки;

32-разрядный указатель на структуру Pnode: родитель узла;

32-разрядный указатель на структуру Pnode: крайний левый потомок;

32-разрядный указатель на структуру Pnode: следующий узел с таким же родителем;

32-разрядный указатель на структуру Pnode: следующий узел с таким же идентификатором элемента;

32-разрядный слово count: значение поддержки

}

Структура Pnode занимает в памяти на 4 байта больше, однако из-за сокращения общего количества узлов при использовании Patricia-tree, общий объем памяти для представления тестового набора записей (табл. 1) в виде FP-tree дерева (рис. 2) больше и составляет 312 байт. В то время как Patricia-tree для того же набора требует 244 байт.

В рассматриваемом наборе (табл. 1) при значении поддержки *min_sup*=2 мы имеем шесть элементов, которые являются частыми, следовательно, наибольшее значение идентификатора элемента не будет больше шести. Тогда вместо массива идентификаторов элементов мы можем использовать только одну 32-разрядную целочисленную переменную key для хранения информации об элементах с одинаковым значением поддержки, а информацию о размере массива идентификаторов элементов исключить из структуры Pnode. В переменной key бит n будет равен единице, если элемент с идентификатором n присутствует в узле, и ноль – в противном случае. При таком варианте Pnode размер Patricia-tree уменьшится до 168 байт. Применение отдельных разрядов для хранения идентификаторов элементов позволило сократить объем оперативной памяти, требуемой Patricia-tree для представления исследуемого набора данных, в сравнении с FP-tree в 1,85 раза. Однако данный подход приемлем только для наборов данных и значений поддержки, при которых количество часто встречающихся элементов будет не более чем 32. Для преодоления указанного ограничения необходимо вновь вернуться к использованию массивов, и очевидными являются несколько способов:

- с каждым узлом связать битовый набор длиной N^{freq}, где N^{freq} – количество частых элементов $i \in I,\ sup\ i \geq min\_sup$. Модификация Pnode: исключить поля «размер массива идентификаторов элементов» и «идентификаторы элементов с одинаковым значением поддержки», добавить 32-разрядный массив слов для хранения N^{freq}-разрядного битового набора. Недостаток: длина массива постоянна, даже если в узле хранится один идентификатор элемента, с узлом будет связан весь битовый набор. Вывод: при больших значениях N^{freq} мы получим избыточное использование памяти, результат заведомо будет хуже, чем при использовании стандартного варианта Pnode.
- вместо всего битового набора хранить только 32-разрядные части, которые сдержат информацию об идентификаторах элементов в узле. Модификация Pnode: добавить массив для хранения индексов 32-разрядных частей. Недостаток: размер поля данных для хранения индекса жестко задается во время компиляции и не может быть динамически изменен во время выполнения. Кроме того, выбор размера поля индекса ограничен 8, 16, и 32(64) разрядными целыми. Вывод: в большинстве случаев мы получаем неэффективное использование памяти для хранения индекса битового набора.

Недостатки вышеописанных способов можно устранить, если хранить индекс битового набора в том же поле данных, что и битовый набор.

Определение 8. Битовой маской идентификаторов элементов называется N-разрядное слово, в котором k разрядов с номерами от N по $N-k+1$ содержат номер битового набора, а разряды с номерами от $N-k$ по 0 – битовый набор. Значение k определяется по формуле

$$k = \min(k'),\quad 2^{k'} * N - k \geq N^{freq}.$$

Для некоторого идентификатора n номер битового набора равен $N^{bit\#} = n \ \text{div}\ k$, все разряды $N-k$-разрядного битового набора равны 0, за исключением разряда с номером $n \bmod k$.

СТРУКТУРА PBnode {

32-разрядное слово: размер массива битовых масок;

32-разрядный указатель на массив 32-разрядных слов: битовые маски идентификаторов элементов с одинаковым значением поддержки;

32-разрядный указатель на структуру PBnode: родитель узла;

32-разрядный указатель на структуру PBnode: крайний левый потомок;

32-разрядный указатель на структуру PBnode: следующий узел с таким же родителем;

32-разрядный указатель на структуру PBnode: следующий узел с таким же идентификатором элемента;

32-разрядный слово count: значение поддержки

}

Модифицировав стандартный вариант Pnode путем введения массива 32-разрядных целых для хранения битовых масок идентификаторов частых элементов, вместо массива идентификаторов, мы получим вариант PBnode, который:

1) не требует выделения дополнительной памяти;

2) позволяет применить битовый набор для хранения идентификаторов частых элементов;

3) позволяет динамически во время выполнения изменять в зависимости от значения N^{freq} количество разрядов, необходимых для хранения индекса битового набора.

Для проверки целесообразности использования предложенной в статье структуры данных PBnode, на основе метода FPclose был создан модифицированный метод поиска замкнутых ЧВНЭ PBclose. Программная реализация PBclose полностью реализует алгоритм работы FPclose, отличием является лишь то, что для хранения исследуемого и проекционных наборов данных реализована поддержка специальной структуры данных PB-tree и ее составляющих PBnode.

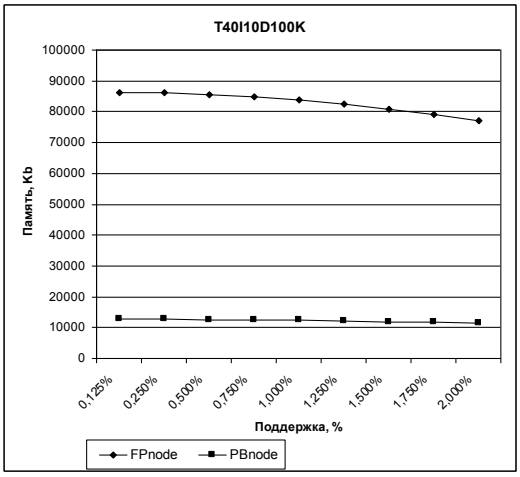
Результаты экспериментальных исследований

В эксперименте использовались написанные на C++ и скомпилированные в среде Microsoft Visual Studio 2005 реализации методов FPclose и PBclose. Все эксперименты проводились на рабочей станции с процессором AMD Athlon 64 X2 5600 (2.9 ГГц) и 1 Гб оперативной памяти под управлением MS Windows XP SP3. В качестве тестовых наборов данных были использованы:

- Синтетические наборы T40I10D100K и T100I20D100K, полученные с помощью синтетического генератора [33]. Каждый из наборов содержит 100000 записей, запись набора T40I10D100K в среднем содержит 40, а T100I20D100K – 100 элементов. Максимальная длина частого набора равна 10 и 20 элементов соответственно.
- Connect. Набор взят с Frequent Itemset Mining Dataset Repository [34], содержит 67557 записей, каждая из которых состоит из 43 элементов.
- Pumsb_star. Набор взят с Frequent Itemset Mining Dataset Repository [34], содержит 49046 записей, каждая из которых состоит из 74 элементов.

Выбор вышеуказанных наборов данных связан с тем фактом, что все они были использованы авторами FPclose для исследования характеристик своего метода. Таким образом, взяв те же самые значения поддержки и учитывая тот факт, что программная реализация PBclose была создана на основе кода FPclose, мы получим условия максимально приближенные к тем, при которых были проведены исследования FPclose [22].

Главным отличием двух методов являются различные структуры данных, которые применяются для представления исследуемого набора в оперативной памяти. Поэтому графики результатов, рис. 4 – 7, по методу FPclose мы обозначали именем используемой структуры данных – FPnode, а по PBclose – PBnode.



$M\ K^{ram} = 6.796$

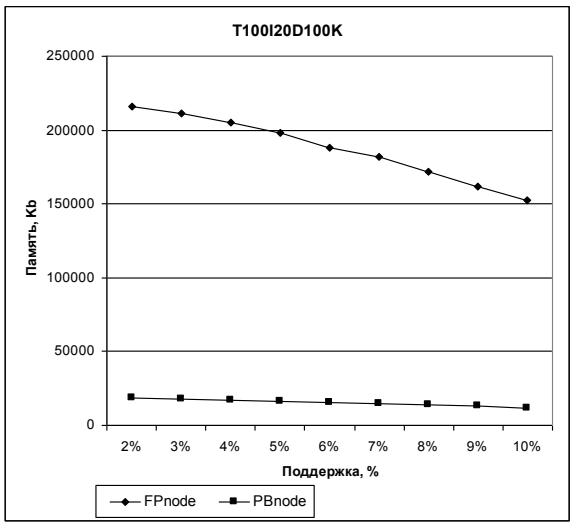
Рис. 4. График зависимости используемой памяти от значения поддержки для тестового набора данных T40I10D100K

Коэффициент K^{ram} равен отношению количества памяти, необходимого для хранения исследуемого набора данных в оперативной памяти в методе FPClose, к объему памяти, требуемого в случае использования модифицированного метода. Среднее значение K^{ram} в конкретном наборе для значений поддержки, используемых в эксперименте, мы обозначили как $M\ K^{ram}$.

Таблица 2

Время построения структур данных в оперативной памяти для набора T40I10D100K

Поддержка	Pbnode	FPnode
0,125%	2,583	2,253
0,250%	2,568	2,191
0,500%	2,443	2,066
0,750%	2,427	1,956
1,000%	2,364	1,925
1,250%	2,208	1,894
1,500%	2,13	1,675
1,750%	1,942	1,675
2,000%	1,849	1,456



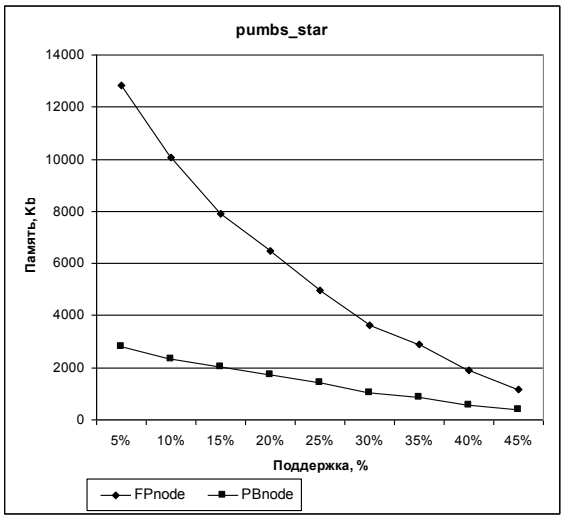
$$M\ K^{ram} = 12.125$$

Рис. 5. График зависимости используемой памяти от значения поддержки для тестового набора данных T100I20D100K

Таблица 3

Время построения структур данных в оперативной памяти для набора T100I20D100K

Поддержка	Pbnode	FPnode
2%	5,831	6,156
3%	5,300	5,469
4%	4,893	5,125
5%	4,253	4,297
6%	3,566	3,859
7%	3,113	3,39
8%	2,785	2,625
9%	2,394	2,563
10%	2,143	2,141



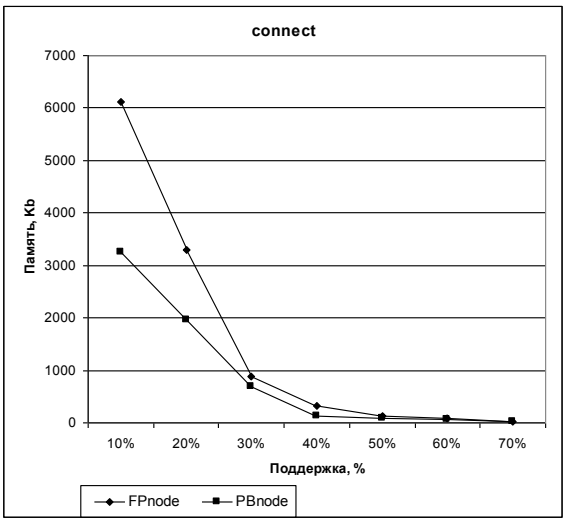
$$M\ K^{ram} = 3.774$$

Рис. 6. График зависимости используемой памяти от значения поддержки для тестового набора данных pumbs_star

Таблица 4

Время построения структур данных в оперативной памяти для набора pumbs_star.

Поддержка	Pbnode	FPnode
5%	0,389	0,270
10%	0,404	0,208
15%	0,326	0,223
20%	0,310	0,207
25%	0,264	0,161
30%	0,217	0,129
35%	0,311	0,052
40%	0,170	0,082
45%	0,154	0,098



$$M\ K^{ram} = 1,836$$

Рис. 7. График зависимости используемой памяти от значения поддержки для тестового набора данных connect

Таблица 5

Время построения структур данных в оперативной памяти для набора connect

Поддержка	Pbnode	FPnode
10%	0,416	0,280
20%	0,353	0,218
30%	0,306	0,202
40%	0,291	0,155
50%	0,275	0,155
60%	0,244	0,155
70%	0,197	0,108
80%	0,166	0,108
90%	0,119	0,046

Модифицированный метод во всех тестовых наборах позволяет достигнуть сокращения объема памяти, необходимого для размещения исследуемого набора данных. Значение коэффициента K^{ram} в каждом конкретном случае зависит от характеристик исследуемого набора данных и порогового зна-

чения поддержки. Как видно из результатов экспериментов (табл. 2 – 5), применение PBnode по сравнению с FPnode связано с временными затратами. Однако отношение времени построения FP-tree ко времени построения PB-tree на порядок меньше, чем аналогичное по сути отношение объемов памяти, выраженное коэффициентом K^{ram}.

Выводы

Научная новизна работы состоит в разработке нового способа представления исследуемого набора данных в методах поиска замкнутых ЧВНЭ, основанных на использовании FP-tree подобных структур данных.

Использование предложенного способа во всех тестовых наборах позволяет достигнуть сокращения объема памяти, необходимого для размещения исследуемого набора данных, в среднем от 1,83 до 12,12 раза.

При этом разница во времени построения структуры в оперативной памяти методом FPclose и модифицированным методом PBclose относительно общего времени процедуры поиска ассоциативных правил является несущественной.

Представленный в статье способ представления исследуемого набора данных может быть использован для модификации методов, основанных на использовании FP-tree подобных структур данных и предназначенных для поиска как всех часто встречающихся наборов элементов, так и их подмножеств: максимальных и замкнутых ЧВНЭ.

В качестве направления дальнейшего исследования рассматривается разработка новых критериев формирования ассоциативных правил, которые бы позволяли уменьшить процент выявления непригодных для дальнейшего использования закономерностей между объектами (сущностями) исследуемой предметной области.

Список литературы

1. Барсегян А.А. Методы и модели анализа данных: OLAP и Data Mining / А.А. Барсегян, М.С. Куприянов, В.В. Степаненко, И.И. Холод. – СПб.: БХВ-Петербург, 2004. – 336 с.

2. Hipp J. Algorithms for Association Rule Mining – a general survey and comparation [Электронный ресурс] / J. Hipp, U. Güntzer, G. Nakhaeizadeh. – Режим доступа: www/ URL: http://www.sigkdd.org/explorations/issues/2-1-2000-06/hipp.pdf – Заголовок с экрана.

3. Kwasnicka P. Discovery of association rules from medical data - classical and evolutionary approaches [Электронный ресурс] / H. Kwasnicka, K. Switalski. – Режим доступа: www/ URL: http://www.iis.pwr.wroc.pl/~kwasnick/download/kwasnickaswitalski.pdf.

4. Delgado M. A Financial Investment Tutoring System [Электронный ресурс] / M. Delgado Calvo-Flores, E. Gibaja Galindo, C. Molina Fernández, J. Núñez Negrillo. – Режим доступа: http://www.formatex.org/micte2006/pdf/741-745.pdf.

5. Au W. Mining Fuzzy Association Rules in A Bank-account Database [Электронный ресурс] / Wai-Ho Au, Keith C.C. Chan. – CiteSeer. Scientific Literature Digital Library and Search Engine. – Режим доступа: www/ URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.62.3731&rep=rep1&type=pdf.

6. Klemettinen M. Rule discovery in telecommunication alarm data [Текст] / M. Klemettinen, H. Mannila, H. Toivonen // Journal of Network and Systems Management. – 1999. – Vol.7(4). – P. 395-423.

7. Gasmi G. Extracting generic basis of association rules from SAGE data [Электронный ресурс] / G. Gasmi, T. Hamrouni, S. Abdelhak, S. Ben Yahia, E. Mephu Nguifo. – Режим доступа: www/ URL: http://lisp.vse.cz/challenge/ecmlpkdd2005/proceedings/03-final.pdf.

8. Shua H. Mining association rules in geographical spatio-temporal data [Электронный ресурс] / H. Shua, X. Zhub, S. Daic. – Режим доступа: www/ URL: http://www.isprs.org/congresses/beijing2008/proceedings/2_pdf/2_WG-II-2/10.pdf.

9. Kou Y. Survey of Fraud Detection Techniques data [Электронный ресурс] / Y. Kou, C-.T. Lu, S. Sirwongwattana, Y-.P. Huang. – Режим доступа: www/ URL: http://www.stttelkom.ac.id/staf/MAB/CS4943/ref/anomaly%20detection%20-%20outlier-fraud/fraud%20detection/00%20 Survey%20of%20Fraud%20Detection%20Techniques%20(2002).pdf.

10. Viglioni M. Methodology for Railway Demand Forecasting Using Data Mining Giovanni [Электронный ресурс] / M.C. Viglioni. – Режим доступа: www/ URL: http://www2.sas.com/proceedings/forum2007/161-2007.pdf.

11. Pasquier N. Efficient Mining Of Association Rules Using Closed Itemset Lattices [Электронный ресурс] / N. Pasquier, Y. Bastide, R. Taouil, L. Lakhal. – CiteSeer. Scientific Literature Digital Library and Search Engine. – Режим доступа: www/ URL: http://citeseerx.ist.psu.edu/ viewdoc/download?doi=10.1.1.23.2928&rep=rep1&type=pdf.

12. Pasquier N. Discovering frequent closed itemsets for association rules / N. Pasquier, Y. Bastide, R. Taouil, L. Lakhal // Proceedings of the 7th International Conference on Database Theory. – 1999. – P. 398-416.

13. Agrawal Rakesh. Fast Algorithms for Mining Association Rules [Электронный ресурс] / Rakesh Agrawal, Ramakrishnan Srikant. – CiteSeer. Scientific Literature Digital Library and Search Engine. – Режим доступа: www/ URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.40.7506&rep=rep1&type=pdf.

14. Orlando S. A Scalable Multi-Strategy Algorithm for Counting Frequent Sets [Электронный ресурс] / S. Orlando, P. Palmerini, R. Perego, F. Silvestri. – CiteSeer. Scientific Literature Digital Library and Search Engine. – Режим доступа: www/ URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.2.3444&rep=rep1&type=pdf.

15. Han J. Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach / Jiawei Han, Jian Pei, Yiwen Yin, Runying Mao // Data Mining and Knowledge Discovery. – 2004. – Vol. 8(1). – P. 53-87.

16. Zaki Mohammed J. Fast Vertical Mining Using Diffsets [Электронный ресурс] / Mohammed J. Zaki, Karam Gouda. – CiteSeer. Scientific Literature Digital Library and Search Engine. – Режим доступа: www/ URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.27.1225&rep=rep1&type=pdf.

17. Goethals Bart. Survey on Frequent Pattern Mining (2003) [Электронный ресурс] / Bart Goethals. – CiteSeer. Scientific Literature Digital Library and Search Engine. – Режим доступа: www/ URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.6.2405&rep=rep1&type=pdf.

18. Yahia S. Frequent closed itemset based algorithm: a thorough structural and analytical survey / S. Ben Yahia,

T. Hamrouni, E. Mephu Nguifo // ACM SIGKDD Explorations Newsletter. – 2006. – P. 93-103.

19. Pei J. CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets [Текст] / J. Pei, J. Han, R. Mao // ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery. – 2000. – P. 21-30.

20. Wang, J. CLOSET+: Searching for the Best Strategies for Mining Frequent Closed Itemsets [Электронный ресурс] / J. Wang, J. Han, J. Pei. – CiteSeer. Scientific Literature Digital Library and Search Engine. – Режим доступа: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.14.3187&rep=rep1&type=pdf.

21. Lucchesse C. DCI Closed: a Fast and Memory Efficient Algorithm to Mine Frequent Closed Itemsets [Электронный ресурс] / C. Lucchesse, S. Orlando, R. Perego. – Режим доступа: www/ URL: http://hpc.isti.cnr.it/~claudio/papers/2004_FIMI_dci_closed.pdf. – Заголовок с экрана.

22. Grahne G. Efficiently Using Prefix-trees in Mining Frequent Itemsets / G. Grahne, J. Zhu. – CiteSeer. Scientific Literature Digital Library and Search Engine. – Режим доступа: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.3.6241&rep=rep1&type=pdf.

23. Uno T. LCM: An Efficient Algorithm for Enumerating Frequent Closed Item Sets [Электронный ресурс] / T. Uno, T. Asai, Y. Uchida и др. – Режим доступа: – http://citeseerx.ist.psu.ed.

24. Uno T. LCM ver. 2: Efficient Mining Algorithms for Frequent/Closed/Maximal Itemsets [Электронный ресурс] / T. Uno, Takeaki Uno, M. Kiyomi, H. Arimura. – Режим доступа: http://research.nii.ac.jp/~uno/papers/0411lcm2.pdf. – Заголовок с экрана.

25. Zaki M.J. CHARM: An Efficient Algorithm for Closed Itemset Mining [Электронный ресурс] / M.J. Zaki, C.-J. Hsiao. – CiteSeer. Scientific Literature Digital Library and Search Engine. – Режим доступа: www/ URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.24.637&rep=rep1&type=pdf.

26. Liu G. AFOPT: An Efficient Implementation of Pattern Growth Approach [Электронный ресурс] / Guimei Liu, Hongjun Lu, Jeffrey Xu Yu. – CiteSeer. Scientific Literature Digital Library and Search Engine. – Режим доступа: www/ URL: http://citeseerx.ist.psu.edu/viewdoc/ download?doi=10.1.1.6.1757&rep=rep1&type=pdf.

27. Liu G. Ascending frequency ordered prefix-tree: Efficient mining of frequent patterns [Текст] / G. Liu, H. Lu, Y. Xu, J. Xu Yu // Proceedings of the Eighth International Conference on Database Systems for Advanced Applications. – 2003. – P. 65-73.

28. ICDM '03. The Third IEEE International Conference on Data Mining. – Режим доступа: www/ URL: http://www.cs.uvm.edu/~icdm/icdm-03.html. – Заголовок с экрана.

29. ICDM '04 The Fourth IEEE International Conference on Data Mining. – Режим доступа: www/ URL: http://icdm04.cs.uni-dortmund.de/. – Заголовок с экрана.

30. Goethals Bart. Advances in Frequent Itemset Mining Implementations: Report on FIMI'03 [Электронный ресурс] / Bart Goethals, M.J. Zaki. – CiteSeer. Scientific Literature Digital Library and Search Engine. – Режим доступа: www/ URL: http://citeseerx.ist.psu.edu/viewdoc/ download?doi=10.1.1.3.136&rep=rep1&type=pdf.

31. http://fimi.cs.helsinki.fi/src/fimi06b.tgz.

32. Pietracaprina A. Mining Frequent Itemsets using Patricia Tries [Электронный ресурс] / A. Pietracaprina, D. Zandolin. – Режим доступа: http://dbdmg.polito.it/twiki/pub/Public/DiskBasedAlgorithm/Patricia-Paper.pdf. – Заголовок с экрана.

33. http://www.almaden.ibm.com/cs/quest/syndata.html.

34. Frequent Itemset Mining Dataset Repository. [Электронный ресурс] – Режим доступа: www/ URL: http://fimi.cs.helsinki.fi/data/. – Заголовок с экрана.

Поступила в редколлегию 9.09.2009

Рецензент: д-р техн. наук, проф. А.А. Кузнецов, Харьковский университет Воздушных Сил им. И. Кожедуба, Харьков.

СПОСІБ ЕФЕКТИВНОГО ПРЕДСТАВЛЕННЯ ДОСЛІДЖУВАНОГО НАБОРУ ДАНИХ В МЕТОДАХ ПОШУКУ АСОЦІАТИВНИХ ПРАВИЛ

О.Л. Стокіпний

У статті даний опис способу представлення вихідного набору даних в методах пошуку замкнутих частих наборів елементів, які базуються на використанні FP-tree подібних структур даних. За результатами експериментів використання запропонованого способу у всіх тестових наборах дозволяє досягти скорочення об'єму пам'яті, необхідного для розміщення досліджуваного набору даних, в середньому від 1.83 до 12.125 разів. Різниця в часі побудови структури в оперативній пам'яті стандартним способом і запропонованим відносно загального часу пошуку асоціативних правил є неістотною.

Ключові слова: *Data Mining, асоціативне правило, частий набір елементів, FP-tree.*

A METHOD OF EFFECTIVE STORAGE OF INVESTIGATED ITEMSET IS IN METHODS OF SEARCH OF ASSOCIATIVE RULES

A.L. Stokipny

In the article description of method of presentation of initial dataset is given in the methods of search of the frequent itemset, based on the use of FP-tree structures of data. On results experiments offered method in all test datasets allows to attain reduction of volume of memory, necessary for placing investigated dataset, on the average from 1.83 to 12.125 time. Difference in time of construction of structure in-memory by a standard method and offered in relation to general time of search of associative rules is to unimportant.

Keywords: *Data Mining, association rule, frequent itemset, FP-tree.*