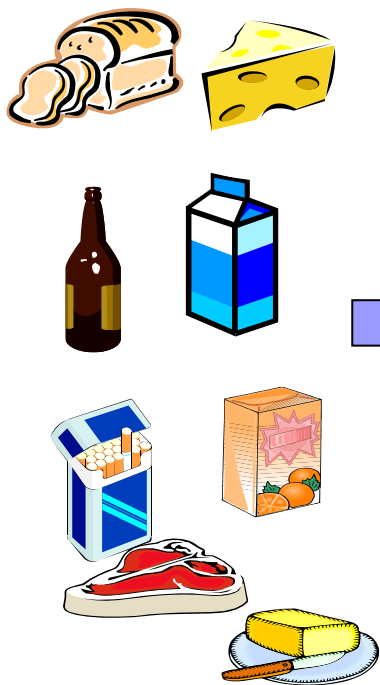


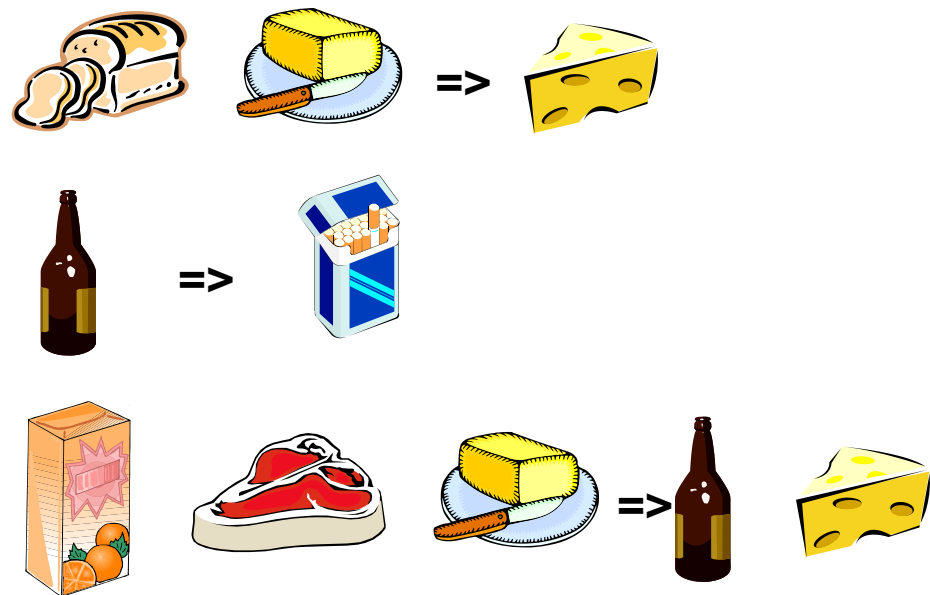
Лекция 19: Поиск ассоциативных правил

Типовая прикладная задача: анализ «корзины покупателя»

Ассортимент
супермаркета



Интересные правила



Задача Определить интересные правила в предпочтениях покупателей при выборе товара

Ассоциативный анализ

■ Правила с семантикой:

- в $\underline{s}\%$ случаях ЕСЛИ верно A_1, A_2, \dots, A_k , ТО с достоверностью \underline{c} будет верно B_1, B_2, \dots, B_m

$$A_1 \wedge A_2 \wedge \dots \wedge A_k \Rightarrow B_1 \wedge B_2 \wedge \dots \wedge B_m$$

- где $A_1, A_2, \dots, A_k, B_1, B_2, \dots, B_m$ – (различные!) предикаты,
- s – поддержка (support), c – достоверность (confidence)

■ Основная задача:

- найти все интересные правила, с заданными ограничениями по s и c (возможно задание дополнительных ограничений на предикаты и сами правила)

■ Основной математический аппарат:

- дискретная математика, математическая логика, комбинаторная оптимизация (на основе метода «ветвей и границ» - вариации полного перебора с отсевом подмножеств допустимых решений, заведомо не содержащих оптимальных решений).

Ассоциативный анализ

- Прикладные задачи:
 - «Экономические»: анализ корзины, маркетинг
 - «Безопасность» и Web usage mining: модели поведения пользователя
 - Text mining: поиск ключевых слов, характеристик и тематик
 - Биоинформатика, медицина
- Задачи анализа:
 - Поиск самих правил
 - Поиск исключений (из правил)
 - Выделение признаков (на основе правил)
 - Классификация и прогнозирование (на базе правил)

Ассоциативный анализ

- Тип моделей:
 - Как правило, «описательный» (descriptive) Data mining => одна из задач - наглядное представление правил
- Тип обучения:
 - «без учителя» (unsupervised) => тренировочный набор не размечен
- Типы правил:
 - **Булевы**
 - Числовые – нужна дискретизация, интервалы как булевы предикаты
 - Иерархические – если определена иерархия для значений атрибутов
 - Временные – как правило, семантика «в с случаях если произошло А и В, то потом случится С и D с вероятностью с»)
 - Пространственные – предикаты определяют пространственные связи между объектами, например «рядом», «далеко» и т.п.

Булевы ассоциативные правила (определение через транзакции)

■ Базовые определения:

- $I = \{i_1, i_2, \dots, i_n\}$ – множество атрибутов/элементов/бинарных признаков;
- D – множество транзакций T , каждая T – множество элементов из I

■ Эпизоды:

- Транзакция T **поддерживает** набор (эпизод) $X \subseteq I$, если $X \subseteq T$
- **Поддержка** эпизода X равна s , если $s\%$ транзакций из D содержат X
- Набор X **часто встречаемый**, если $\text{support}(X) = p(X)$ выше порога

■ Правило:

- **Ассоциативное правило** - импликация $X \Rightarrow Y$, $X \subseteq I$, $Y \subseteq I$ и $X \cap Y = \emptyset$
- Правило $X \Rightarrow Y$ имеет **поддержку (support) s** , если $s\%$ транзакций из D содержат X и Y : $\text{support}(X \Rightarrow Y) = p(X, Y)$
- Правило $X \Rightarrow Y$ имеет **достоверность (confidence) c** , если $c\%$ транзакций из D , содержащих X , также содержат Y :

$$\text{confidence}(X \Rightarrow Y) = p(Y|X) = \frac{p(X, Y)}{p(X)} = \frac{\text{support}(X \cup Y)}{\text{support}(X)}$$

- Правило **интересное**, если превышены заданные пороги на достоверность и поддержку

Булевы ассоциативные правила (определение через признаки)

■ Базовые определения:

- X – пространство объектов;
- $X^l = \{x_1, \dots, x_l\} \subset X$ – обучающая выборка
- $\mathcal{F} = \{f_1, \dots, f_n\}, f_j: X \rightarrow \{0,1\}$ – бинарные признаки (items)

■ Эпизоды:

- Каждому подмножеству $\varphi \subseteq \mathcal{F}$ соответствует конъюнкция
$$\varphi(x) = \bigwedge_{f \in \varphi} f(x), x \in X$$
- Если $\varphi(x) = 1$, то «признаки из φ совместно встречаются у x ».
- Частота встречаемости (поддержка, support) φ в выборке X^l

$$v(\varphi) = \frac{1}{l} \sum_{i=1}^l \varphi(x_i)$$

- Если $v(\varphi) \geq \delta$, то «набор φ частый» (frequent itemset)
- Параметр δ – минимальная поддержка, MinSupp

Булевы ассоциативные правила (определение через признаки)

- **Интересное** ассоциативное правило $\varphi \rightarrow y$ – это пара непересекающихся наборов $\varphi, y \subseteq \mathcal{F}$ таких, что:
 - наборы φ и y совместно часто встречаются, $v(\varphi \cup y) \geq \delta$;
 - если встречается φ , то часто встречается также и y ,
$$v(y|\varphi) \equiv \frac{v(\varphi \cup y)}{v(\varphi)} \geq \kappa$$
, где $v(y|\varphi)$ – значимость (confidence) правила.
 - Параметр κ – минимальная значимость, MinConf.
- **Логическая** интерпретация:
 - Предикат $\varphi(x)$ – логическая закономерность класса $c \in Y$
$$\text{support}(\varphi) = \frac{p(\varphi)}{l} \geq \delta; \quad \text{confidence}(\varphi) = \frac{p(\varphi)}{p(\varphi) + n(\varphi)} \geq \kappa$$
 - $p(\varphi) = \#\{x_i \in X^l: \varphi(x_i) = 1 \text{ и } y(x_i) = c\}$ +примеры класса c
 - $n(\varphi) = \#\{x_i \in X^l: \varphi(x_i) = 1 \text{ и } y(x_i) \neq c\}$ -примеры класса c
 - Для « $\varphi \Rightarrow y$ » возьмем целевой признак $y(x) = \bigwedge_{f \in y} f(x)$.
 - Тогда $v(\varphi \cup y) \equiv \text{Supp}_1(\varphi) \geq \delta$; $\frac{v(\varphi \cup y)}{v(\varphi)} \equiv \text{Conf}_1(\varphi) \geq \kappa$

Пример

D=

t	Хлеб	Кефир	Пиво	Чипсы
1	1	0	0	0
2	1	1	0	0
3	0	1	1	1
4	0	1	1	1
5	1	1	0	0
6	1	0	1	0
7	1	1	1	1
8	1	0	0	0
9	0	0	1	0
10	0	0	1	0

$I = \{\text{Хлеб, Кефир, Пиво, Чипсы}\}$

$\text{supp}(\text{Хлеб}) = 60\%$

$\text{supp}(\text{Кефир}) = 50\%$

$\text{supp}(\text{Пиво}) = 60\%$

$\text{supp}(\text{Чипсы}) = 30\%$

Пример правила: $\text{Пиво} \Rightarrow \text{Чипсы}$

$\text{supp}(\text{П} \Rightarrow \text{Ч}) = 30\%$

$\text{conf}(\text{П} \Rightarrow \text{Ч}) = 50\%$

Задача: Найти правила с параметрами:

$\text{minsupp} = 30\%$,

$\text{minconf} = 60\%$

Критика Support и Confidence

■ Пример: (Aggarwal & Yu, PODS98)

- Среди 5000 студентов
 - 3000 играют баскетбол
 - 3750 любят черный хлеб
 - 2000 и то и другое

	basketball	not basketball	sum(row)
bread	2000	1750	3750
not bread	1000	250	1250
sum(col.)	3000	2000	5000

- $basketball \Rightarrow bread$ [40%, 66.7%] вводит в заблуждение, поскольку процент любителей хлеба 75% выше support 66.7%.
- $basketball \Rightarrow not\ bread$ [20%, 33.3%] более полезное, хотя support and confidence ниже

■ Пример:

- X и Y: положительно коррелированы,
- X и Z, отрицательно коррелированы
- support и confidence больше у $X \Rightarrow Z$

X	1	1	1	1	0	0	0	0
Y	1	1	0	0	0	0	0	0
Z	0	1	1	1	1	1	1	1

Rule	Support	Confidence
$X \Rightarrow Y$	25%	50%
$X \Rightarrow Z$	37,50%	75%

■ Нужна новая мера зависимости

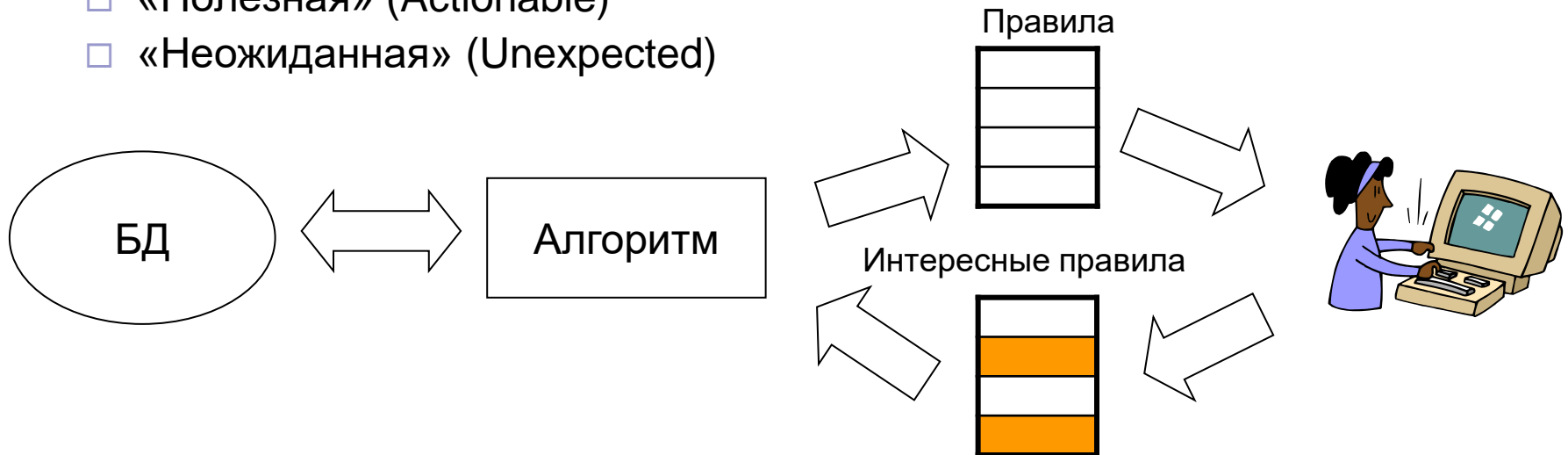
Интересность

■ Объективная:

- $\text{support}(X \Rightarrow Y), \text{confidence}(X \Rightarrow Y), \text{generality}(X \Rightarrow Y) = P(Y)$
- $\text{lift}(X \Rightarrow Y) = \frac{P(Y|X)}{P(Y)} = \frac{\text{confidence}(X \Rightarrow Y)}{\text{generality}(X \Rightarrow Y)}$
- $\text{RI}(X \Rightarrow Y) = P(Y|X) - P(Y) = \text{confidence}(X \Rightarrow Y) - \text{generality}(X \Rightarrow Y)$
- $J(X \Rightarrow Y) = P(Y) \left[P(Y|X) \log_2 \frac{P(Y|X)}{P(Y)} + (1 - P(Y|X)) \log_2 \frac{1 - P(Y|X)}{1 - P(Y)} \right]$

■ Субъективная (на основе информации, заданной экспертом)

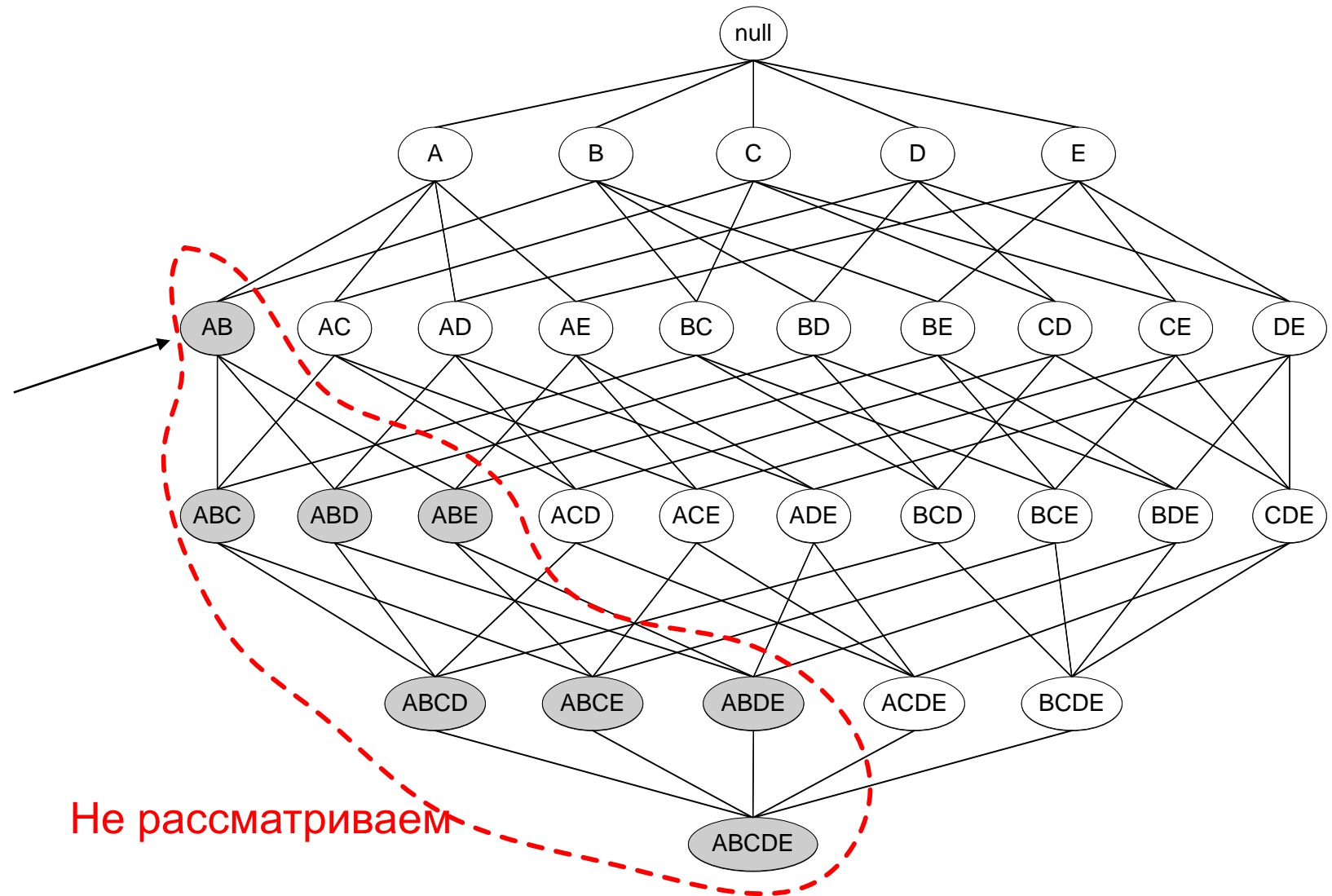
- «Полезная» (Actionable)
- «Неожиданная» (Unexpected)



Алгоритм Apriori

- Основной принцип (анти-монотонность):
 - Любое подмножество часто встречаемого набора является часто встречаемым набором
- Формально:
 - Поддержка любого набора элементов не может превышать минимальной поддержки всех его подмножеств
 - Необходимое условие частой встречаемости k -элементного набора – частая встречаемость всех его $(k-1)$ -элементных подмножеств
 - В примере: $\text{supp}(\{\text{Хлеб, Кефир, Чипсы}\})$ не больше чем $\text{supp}(\{\text{Хлеб, Кефир}\})$, $\text{supp}(\{\text{Хлеб, Чипсы}\})$, $\text{supp}(\{\text{Кефир, Чипсы}\})$, $\text{supp}(\{\text{Кефир}\})$, $\text{supp}(\{\text{Чипсы}\})$, $\text{supp}(\{\text{Хлеб}\})$
- Этапы алгоритма:
 - Генерация множества часто встречаемых наборов ($\text{supp} \geq \text{minsupp}$): метод «ветвей и границ» - направленный перебор от простых (коротких) наборов к сложным (длинным) с отсечением
 - Генерация правил по найденным наборам ($\text{conf} \geq \text{minconf}$)

Идея метода ветвей и границ для Apriori



Пример генерации кандидатов с отсечением

- $L_3 = \{abc, abd, acd, ace, bcd\}$
- Join: $L_3 * L_3$
 - $abcd = abc + abd$
 - $acde = acd + ace$
- Pruning:
 - $acde$ удален, т.к. ade не в L_3
- $C_4 = \{abcd\}$

Алгоритм Apriori (формально)

- Вход: X^l – обучающая выборка; $\delta = MinSupp$; $\kappa = MinConf$;
- Выход: $R = \{(\varphi, y)\}$ – список ассоциативных правил;
- Множество всех частых исходных признаков:
 - $G_1 := \{f \in \mathcal{F} | v(f) \geq \delta\}$;
- Для всех $j = 2, \dots, n$
 - Множество всех частых наборов мощности j :
$$G_j := \{\varphi \cup \{f\} | \varphi \in G_{j-1}, f \in G_1 \setminus \varphi, v(\varphi \cup \{f\}) \geq \delta\};$$
 - Если $G_j = \emptyset$, то Выход из цикла по j ;
- $R := \emptyset$;
- Для всех $\psi \in G_j, j = 2, \dots, n$ AssocRules(R, ψ, \emptyset);

Пример

D=

t	Хлеб	Кефир	Пиво	Чипсы
1	1	0	0	0
2	1	1	0	0
3	0	1	1	1
4	0	1	1	1
5	1	1	0	0
6	1	0	1	0
7	1	1	1	1
8	1	0	0	0
9	0	0	1	0
10	0	0	1	0

Построение L1

$\text{supp}(\text{Хлеб}) = 60\%$

$\text{supp}(\text{Кефир}) = 50\%$

$\text{supp}(\text{Пиво}) = 60\%$

$\text{supp}(\text{Чипсы}) = 30\%$

L1 = {{X}, {K}, {П}, {Ч}}

Построение L2

{X, K}, {X, П}, {X, Ч}

{K, П}, {K, Ч}, {П, Ч}

$\text{supp}(\{X, K\}) = 30\%$

$\text{supp}(\{X, П\}) = 20\%$

$\text{supp}(\{X, Ч\}) = 10\%$

$\text{supp}(\{K, П\}) = 30\%$

$\text{supp}(\{K, Ч\}) = 30\%$

$\text{supp}(\{П, Ч\}) = 30\%$

L2={{X,K}, {K,П}, {K,Ч}, {П,Ч}}

Пример

D=

t	Хлеб	Кефир	Пиво	Чипсы
1	1	0	0	0
2	1	1	0	0
3	0	1	1	1
4	0	1	1	1
5	1	1	0	0
6	1	0	1	0
7	1	1	1	1
8	1	0	0	0
9	0	0	1	0
10	0	0	1	0

$L2 = \{\{X, K\}, \{K, П\}, \{K, Ч\}, \{П, Ч\}\}$

Формируем L3

$\{K, П, Ч\}$

$\text{supp}(\{K, П, Ч\}) = 30\%$

L3 = $\{\{K, П, Ч\}\}$

Результат=

$\{\{X\}60\%, \{K\}50\%, \{П\}60\%, \{Ч\}30\%,$
 $\{X, K\}30\%, \{K, П\}30\%, \{K, Ч\}30\%,$
 $\{П, Ч\}30\%, \{K, П, Ч\}30\%\}$

Генерация правил

- Дано и условия:

- $\text{confidence}(X \Rightarrow Y) = \frac{\text{support}(X, Y)}{\text{support}(X)}$
- Должно выполняться $\text{confidence}(X \Rightarrow Y) \geq \kappa$
- все $\text{support}(X, Y) \geq \delta$ и известны с 1-го этапа – генерации эпизодов

- Принцип:

- Если правило $\{A\} \Rightarrow \{B, C\}$ интересно, то и $\{A, B\} \Rightarrow \{C\}$ интересно

- Доказательство:

$$\text{confidence}(\{A\} \Rightarrow \{B, C\}) = \frac{\text{support}(\{A, B, C\})}{\text{support}(\{A\})} \geq \kappa$$

т.к. $\text{support}(\{A, B\}) \leq \text{support}(\{A\})$, то:

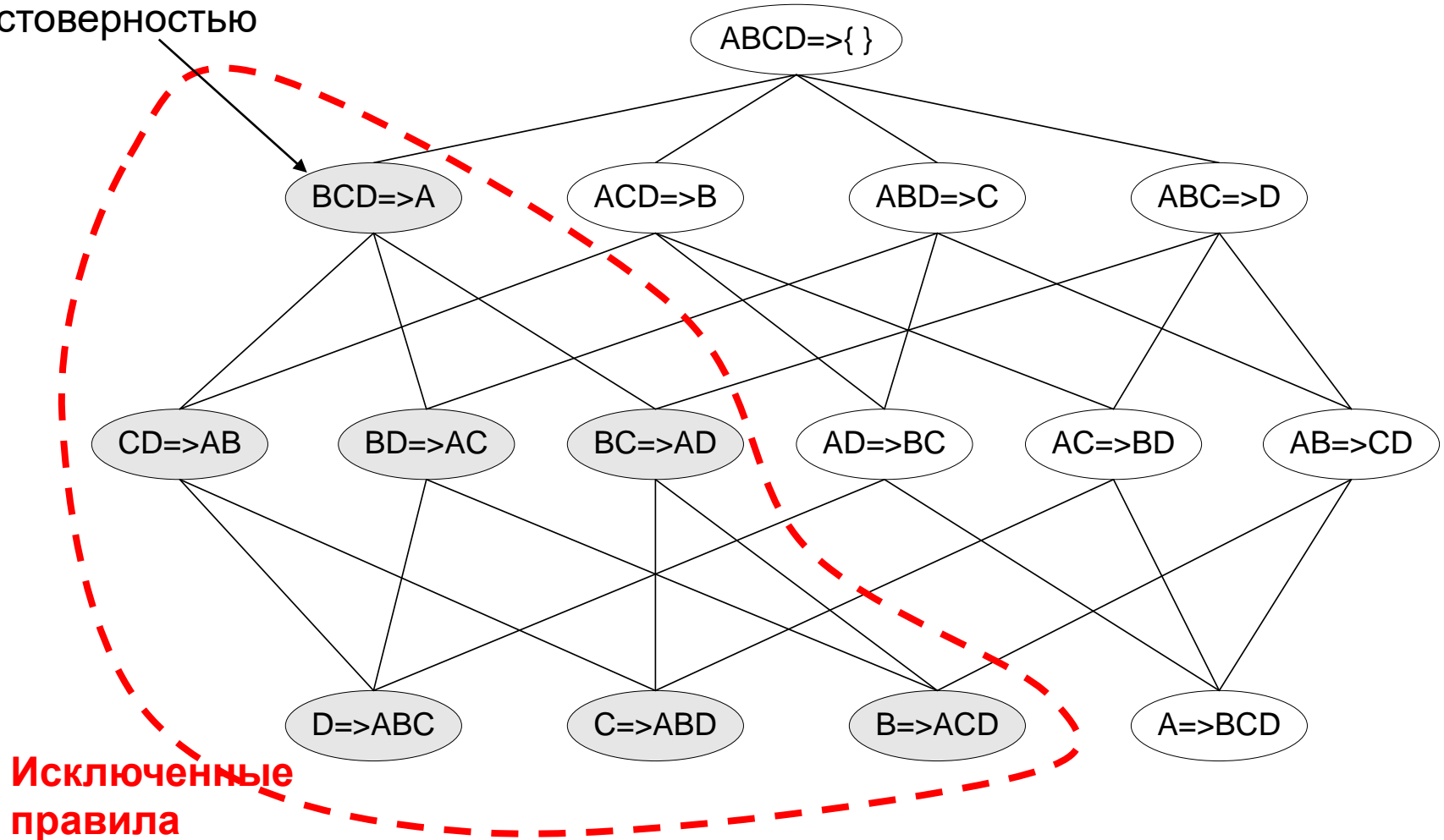
$$\text{confidence}(\{A, B\} \Rightarrow \{C\}) = \frac{\text{support}(\{A, B, C\})}{\text{support}(\{A, B\})} \geq \frac{\text{support}(\{A, B, C\})}{\text{support}(\{A\})} \geq \kappa$$

- Алгоритм:

- Для каждого часто встречаемого набора проверять правила на интересность, начиная со случая, когда в правой части правила находится один атрибут и постепенно добавлять/убавлять атрибуты в/из правую/левую часть(и).

Метод ветвей и границ для генерации правил

Правило с низкой достоверностью



Выделение ассоциативных правил

- Этап 2. Простой рекурсивный алгоритм, выполняемый быстро, как правило, полностью в оперативной памяти.
- Функция $\text{AssocRules}(R, \varphi, y)$
 - Вход: (φ, y) – ассоциативное правило;
 - Выход: R – список ассоциативных правил;
 - Для всех $f \in \varphi: id_f > \max_{g \in y} id_g$ (чтобы избежать повторов y)
 - $\varphi' := \varphi \setminus \{f\}; y' := y \cup \{f\};$
 - если $v(y'|\varphi') \geq \kappa$ то добавить ассоциативное правило (φ', y') в список R ;
 - Если $|\varphi'| > 1$ то $\text{AssocRules}(R, \varphi', y')$;
 - id_f – порядковый номер признака f в $\mathcal{F} = \{f_1, \dots, f_n\}$

Пример

D=

t	Хлеб	Кефир	Пиво	Чипсы
1	1	0	0	0
2	1	1	0	0
3	0	1	1	1
4	0	1	1	1
5	1	1	0	0
6	1	0	1	0
7	1	1	1	1
8	1	0	0	0
9	0	0	1	0
10	0	0	1	0

Наборы:

- $\{X\}60\%, \{K\}50\%, \{П\}60\%, \{Ч\}30\%,$
 $\{X, K\}30\%, \{K, П\}30\%, \{K, Ч\}30\%, \{П, Ч\}30\%,$
 $\{K, П, Ч\}30\%$

Правила:

$\text{conf}(\{X\} \Rightarrow \{K\}) = 50\%$
 $\text{conf}(\{K\} \Rightarrow \{X\}) = 60\%$
 $\text{conf}(\{K\} \Rightarrow \{П\}) = 60\%$
 $\text{conf}(\{П\} \Rightarrow \{K\}) = 50\%$
 $\text{conf}(\{K\} \Rightarrow \{Ч\}) = 60\%$
 $\text{conf}(\{Ч\} \Rightarrow \{K\}) = 100\%$
 $\text{conf}(\{П\} \Rightarrow \{Ч\}) = 50\%$
 $\text{conf}(\{Ч\} \Rightarrow \{П\}) = 100\%$
 $\text{conf}(\{K, П\} \Rightarrow \{Ч\}) = 100\%$
 $\text{conf}(\{K\} \Rightarrow \{П, Ч\}) = 60\%$
 $\text{conf}(\{П\} \Rightarrow \{K, Ч\}) = 50\%$
 $\text{conf}(\{K, Ч\} \Rightarrow \{П\}) = 100\%$
 $\text{conf}(\{Ч\} \Rightarrow \{K, П\}) = 100\%$
 $\text{conf}(\{П, Ч\} \Rightarrow \{K\}) = 100\%$

Недостатки Apriori

- Суть алгоритма Apriori:
 - Использовать часто встречаемые наборы размера $(k - 1)$ для генерации кандидатов часто встречаемых наборов размера k
 - Использовать dbscan и сравнения подмножеств атрибутов для расчета поддержки кандидатов
- Слабое место – генерация кандидатов
 - Огромное число кандидатов: 10^4 1-элементных наборов приводят к 10^7 2-элементным наборам, если надо найти наборы размера 100 $\{a_1, a_2, \dots, a_{100}\}$, нужно сгенерировать $2^{100} \approx 10^{30}$ кандидатов.
 - Множественные dbscan: $(n + 1)$ сканирований, где n - длина наибольшего набора
- Пути решения:
 - Хэш-деревья для хранения наборов и счетчиков поддержки
 - Удаление неинформативных транзакций из базы
 - Разбиение базы и sampling - набор будет часто встречаемым, если он часто встречаемый на каком-то подмножестве транзакций, но: необходима оценка полноты и достоверности

Поиск частых наборов без кандидатов

- Основная задача, решаемая методом **Frequent-Pattern tree** :
 - «сжать» информацию о транзакциях и представить в «компактном» виде с быстрым поиском частых наборов
 - уйти от частых сканирований БД, не генерировать кандидатов, а искать их динамически по структуре FP-tree
 - стратегия «разделяй и властвуй»: декомпозиция задачи поиска на более мелкие подзадачи – рекурсивное построение «пути» частых наборов в FP-tree дереве
- Свойства и требования к структуре:
 - «сжатая» информация для поиска наборов должна быть полной
 - размер вспомогательных структур не должен превосходить размер БД
 - не должно быть несодержательной информации, например
 - при поиске обратная упорядоченность по частоте наборов и атрибутов – более часто встречаемые атрибуты с большой вероятностью являются частью частых наборов

Построение FP-tree

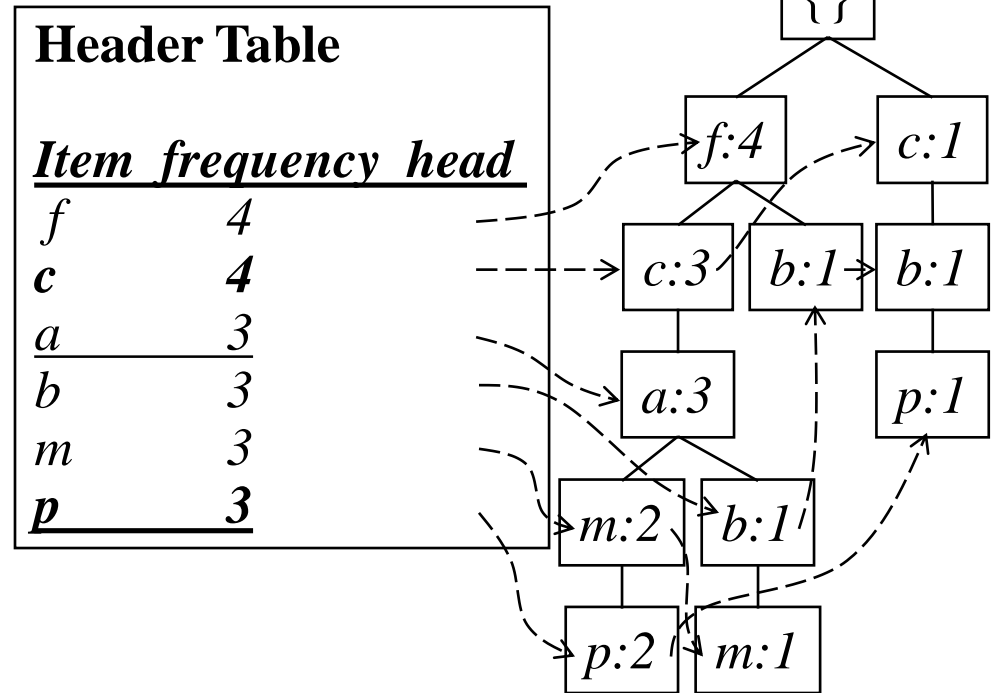
<i>TID</i>	<i>Items</i>	<i>frequent items</i>
100	{ <i>f, a, c, d, g, i, m, p</i> }	{ <i>f, c, a, m, p</i> }
200	{ <i>a, b, c, f, l, m, o</i> }	{ <i>f, c, a, b, m</i> }
300	{ <i>b, f, h, j, o</i> }	{ <i>f, b</i> }
400	{ <i>b, c, k, s, p</i> }	{ <i>c, b, p</i> }
500	{ <i>a, f, c, e, l, p, m, n</i> }	{ <i>f, c, a, m, p</i> }

min_support = 0.5

Шаги:

1. Первое сканирование БД и построение частых 1-наборов
2. Обратная сортировка по частоте
3. Второе сканирование и построение FP-tree

По заголовочной таблице можно проверить является эпизод частым: {*p, a*}? {*p, c*}?



Префиксное FP-дерево

- В каждой вершине v дерева T задается тройка $\langle f_v, c_v, S_v \rangle$:
 - Признак $f_v \in \mathcal{F}$;
 - Множество дочерних вершин $S_v \subset T$;
 - Счетчик поддержки $c_v = lv(\varphi_v)$ набора $\varphi_v = \{f_u: u \in [v_0, v]\}$, где $[v_0, v]$ – путь от корня дерева v_0 до вершины v .
- Обозначения:
 - $V(T, f) = \{v \in T: f_v = f\}$ – все вершины признака (уровня) f .
 - $\mathcal{C}(T, f) = \sum_{v \in V(T, f)} c_v$ – сумма счетчиков поддержки признака f .
- Свойства FP-дерева T , построенного по всей выборке X^l :
 - $\frac{1}{l} \mathcal{C}(T, f) = v(f)$ – поддержка признака f .
 - T содержит информацию о $v(\varphi)$ всех частых наборов φ .

Алгоритм FP-growth

- Вход: X^l – обучающая выборка;
- Выход: FP-дерево T ; $\langle f_v, c_v, S_v \rangle$ для всех вершин $v \in T$;
- Упорядочить признаки $f \in \mathcal{F}$: $v(f) \geq \delta$ по убыванию $v(f)$;
- ЭТАП 1: Построение FP-дерева T по выборке X^l
- Для всех $i := 1, \dots, l$
 - $v := v_0$
 - Для всех $f \in \mathcal{F}$ таких, что $f(x_i) = 1$, по убыванию $v(f)$
 - Если нет дочерней вершины $u \in S_v$: $f_u = f$ то
 - Создать новую вершину u ; $S_v := S_v \cup \{u\}$;
 - $f_u := f$; $c_u := 0$; $S_u := \emptyset$;
 - $c_u := c_u + 1$; $v := u$;
- ЭТАП 2: Рекурсивный поиск частых наборов по FP-дереву T
- FP-find(T, \emptyset, \emptyset);

Поиск частых наборов с FP-tree

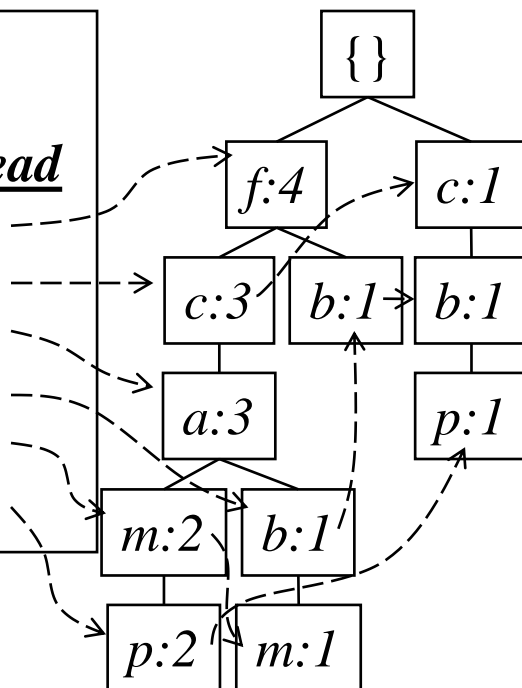
■ Метод:

- Для каждого элемента (начиная с самых редких) найти его **условный базовый набор**
- На основе условного базового набора построить новое **условное FP-tree поддерево** для каждого элемента, рассматривая каждый путь как отдельную транзакцию
- **Повторить процесс** для элементов каждого вновь созданного условного FP-tree поддерева
- До тех пор пока результирующее FP-tree не будет **пусто** или не будет содержать **единственный путь**
- **Единственный путь** генерирует все комбинации подпутей, каждый из которых есть **частый набор**

Шаг 1: От FP-tree к условному базовому набору

- Для каждого элемента проход FP-tree «вверх» по дугам с запоминанием «условного» пути и его поддержки
- В результате с каждым элементом связан условный базовый набор (набор возможных путей к вершине с поддержкой)

Header Table		
<i>Item</i>	<i>frequency</i>	<i>head</i>
<i>f</i>	4	
<i>c</i>	4	
<i>a</i>	3	
<i>b</i>	3	
<i>m</i>	3	
<i>p</i>	3	



Conditional pattern bases

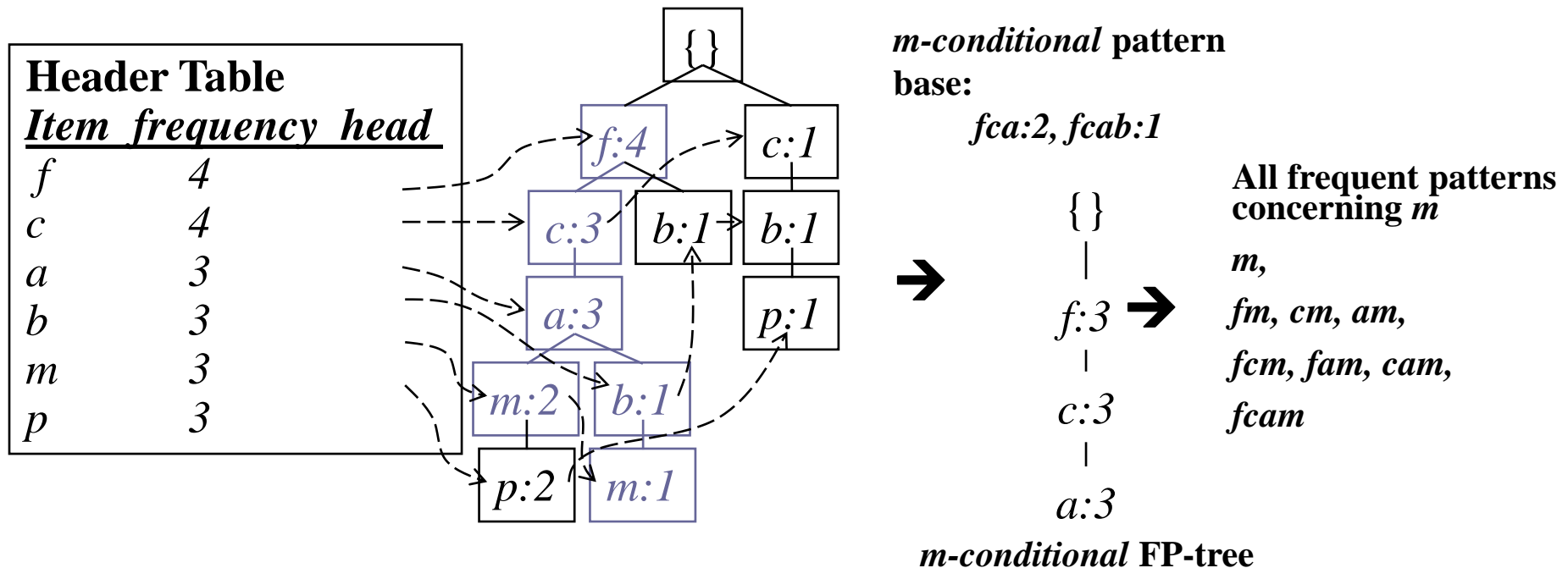
<i>item</i>	<i>cond. pattern base</i>
<i>c</i>	<i>f:3</i>
<i>a</i>	<i>fc:3</i>
<i>b</i>	<i>fca:1, f:1, c:1</i>
<i>m</i>	<i>fca:2, fcab:1</i>
<i>p</i>	<i>fcam:2, cb:1</i>

Свойства условного FP-tree (без доказательства)

- Свойство «узел-связь»:
 - Для каждого частого элемента a , все возможные частые наборы, содержащие его, могут быть получены обходом по пути «узел-связь» от заголовочного элемента к корню FP-tree
- Свойство префикса:
 - Для поиска частых наборов для узла a в пути P , необходимо рассматривать только префикс под-пути от a в P , его поддержка должна быть равна поддержке узла a .

Шаг 2: Построение условного FP-tree

- Для каждого условного базового набора построить условное FP-tree, содержащее только пути из базового набора



Поиск частых наборов по условным базовым наборам

Item	Условный базовый набор	Условное FP-tree
p	{(fcam:2), (cb:1)}	{(c:3)} p
m	{(fca:2), (fcab:1)}	{(f:3, c:3, a:3)} m
b	{(fca:1), (f:1), (c:1)}	Empty
a	{(fc:3)}	{(f:3, c:3)} a
c	{(f:3)}	{(f:3)} c
f	Empty	Empty

Шаг 3: Рекурсивная обработка условного FP-tree

$\{\}$
 $|$
 $f:3$
 $|$
 $c:3$
 $|$
 $a:3$

m-conditional FP-tree

Условный базовый набор для "am": (fc:3)

$\{\}$
 $|$
 $f:3$
 $|$
 $c:3$

am-conditional FP-tree

Условный базовый набор для "cm": (f:3)

$\{\}$
 $|$
 $f:3$

cm-conditional FP-tree

Условный базовый набор для "cam": (f:3)

$\{\}$
 $|$
 $f:3$

cam-conditional FP-tree

Единственный путь в FP-tree

- Предположим в FP-tree есть единственный путь Р
- Полное множество частых наборов из Т могут быть получены перебором всех возможных комбинаций подпутей из Р

$\{\}$
|
 $f:3$
|
 $c:3$
|
 $a:3$



**All frequent patterns
concerning m**

m ,
 fm , cm , am ,
 fcm , fam , cam ,
 $fcam$

m-conditional FP-tree

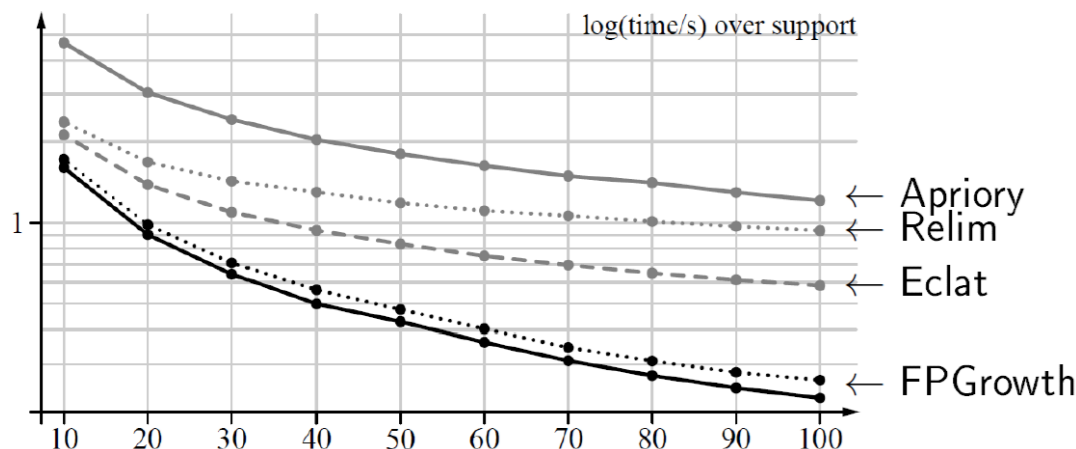
Этап 2: Рекурсивный поиск частых наборов по FP-дереву

- $\text{FP-find}(T, \varphi, R)$ находит по FP-дереву T все частые наборы, содержащие частый набор φ , и добавляет их в список R .
- Две идеи эффективной реализации FP-find :
 - Вместо T достаточно передать условное FP-дерево $T|\varphi$, это FP-дерево, порождаемое подвыборкой $\{x_i \in X^l: \varphi(x_i) = 1\}$
 - Будем добавлять в φ только те признаки, которые находятся выше в FP-дереве. Так мы переберем все подмножества $\varphi \subseteq \mathcal{F}$
- Функция $\text{FP-find}(T, \varphi, R)$
 - Вход: FP-дерево T , частый набор φ , список наборов R ;
 - Выход: Добавить в R все частые наборы, содержащие φ ;
 - Для всех $f \in \mathcal{F}: V(T, f) \neq \emptyset$ по уровням снизу вверх, если $C(T, f) \geq l\delta$:
 - Добавить частый набор $\varphi \cup \{f\}$ в список R ;
 - $T' := T|f$ – условное FP-дерево, найти по T' все частые наборы, включающие φ и f : $\text{FP-find}(T', \varphi \cup \{f\}, R)$;

Преимущества FP-tree перед Apriori

■ Экспериментально:

- FP-tree значительно быстрее Apriori



■ Причина

- Нет генерации и проверки кандидатов, нет повторяющихся сканирований БД, используется компактная структура для поиска частых наборов и расчета поддержки, основные операции – суммирование и построение дерева

Использование ограничений

- Проблема итеративного анализа больших объемов данных:
 - невозможно без использования ограничений
- Типы ограничений:
 - стандартные: на support и confidence
 - на меры объективной интересности
 - на выборку «горизонтально» - подмножества транзакций
 - на выборку «вертикально» - подмножества атрибутов
 - на значения отдельных атрибутов (с точки зрения алгоритма аналогично «вертикальному»)
 - шаблоны правил для поиска – метаправила (задаются экспертом, учитываются методом «ветвей и границ» при генерации наборов и правил из них, сокращается перебор)
 - шаблоны «неинтересных» правил (поиск «неожиданных» правил, нарушающих шаблон) – для оценки субъективной интересности