Методы хранения MOLAP-данных

Докладчик: Юрий Кудрявцев
Yuri.Kudryavcev@gmail.com

План Доклада

- Общие понятия OLAP. Виды запросов к OLAP-данных. MOLAP, HOLAP, ROLAP. 5 минут
- Классификация методов хранения.
- Синтаксические алгоритмы. DWARF. 15 минут.
- Семантические алгоритмы. Quotient Cube. 15 минут
- Аппроксимирующие алгоритмы. BUC, Wavelets. 5 минут
- Визуализация OLAP данных. Модель CPM. 5 минут
- Математическая модель OLAP кубов. 15 минут

Общие понятия OLAP.

- OLAP (OnLine Analytical Processing) – инструмент интерактивного анализа данных пользователем.
- Задачи OLAP-средств:
- ✓ Предоставление больших объемов данных пользователю для анализа
- ✓ Большая скорость работы (время обработки запроса не должно превышать 3 секунд)
- ✓ Удобная модель представления данных (многомерные кубы). Интуитивно понятные интерфейсы работы

12 признаков OLAP.

- **Многомерная концепция данных**. OLAP оперирует CUBE данными, которые являются многомерными массивами данных. Число измерений OLAP кубов не ограничено.
- **Прозрачность.** OLAP системы должны опираться на открытые системы, поддерживающие гетерогенные источники данных.
- **Доступность.** OLAP системы должны представлять пользователю единую логическую схему данных.
- **Постоянная скорость выполнения запросов.** Производительность не должна падать при росте числа измерений.
- **Клиент\сервер архитектура.** Системы должны базироваться на открытых, модульных системах.
- **Различное число измерений.** Системы не должны ограничиваться 3хмерной моделью представления данных. Причем измерения должны быть эквивалентны по применению любых функций.

12 признаков OLAP.

- **Динамическое представление разреженных матриц.** Идея относится к «нулям» в реляционных базах данных и сжатию больших файлов, «разреженная матрица» - матрица, не каждая ячейка которой содержит данные. OLAP системы должны содержать средства хранении и обработки больших объемов данных.
- **Многопользовательская поддержка.** OLAP системы должны поддерживать многопользовательский режим работы.
- **Неограниченные многомерные операции.** Аналогично, требованию о различном числе измерений : все измерения считаются равными и многомерные операции не должны накладывать ограничений на отношения между ячейками.
- **Интуитивно понятные инструменты манипулирование данными.** В идеале, пользователи не должны пользоваться различными усложненными меню и прочим, чтобы сформулировать многоуровневые запросы.
- **Гибкая настройка конечных отчетов.** Пользователи должны иметь возможность видеть только то, что им необходимо, причем все изменения данных должны немедленно отображаться в отчетах.
- **Отсутствие ограничений на количество измерений и уровней агрегации данных**

Пример куба. Фактическая таблица

Регион	Продукт	Время	Продажи
R1	Книги	Весна	9
R2	Еда	Осень	3
R1	Книги	Осень	6

Создание куба.

- В каждое измерение добавляется значение ALL, агрегирующее все остальные значения измерения
- Выбирается агрегирующая функция
- Согласно выбранной функции вычисляются значения для всех возможных кортежей
- Получается OLAP куб для анализа.
- Для таблицы из примера в формате BSF (Binary Storage FootPrint) куб займет 27 кортежей

Виды запросов к OLAP-данным

Точечные запросы (Point Queries)

```
SELECT
   регион,продукт,время,SUM(продажи)
FROM Продажи
WHERE (регион = R1) AND (продукт = книги)
   AND (время = весна)
```

Результат: ячейка (R1,книги,весна)

Виды запросов к OLAP-данным.

Интервальные запросы (Range Queries)

SELECT регион, продукт, SUM(продажи)

FROM Продажи

WHERE

((регион = R1)OR((регион = R2))

AND

(продукт = книги)

GROUP BY регион, продукт

Результат: продажи книг в R1 и R2

Виды запросов к OLAP-данным.

Обратные запросы (Iceberg Queries)

SELECT регион, продукт, время, AVG(продажи)

FROM Продажи

CUBE BY регион, продут, время HAVING AVG(продажи) >= 6

Классификация OLAP-средств.

- ROLAP (Relational OLAP) – все данные куба (фактическая таблица + агрегаты) хранятся в реляционной таблице
- HOLAP (Hybrid OLAP) – фактические данные хранятся в реляционной таблице, агрегаты – в виде многомерных структур (многомерной БД)
- MOLAP (Multidimensional OLAP) – и фактические данные, и агрегаты хранятся в многомерной БД

Классификация методов хранения MOLAP-данных

Синтаксические – преобразующие только схему хранения данных

Семантические – алгоритмы, преобразующие структуру куба, например, меняющие roll-up, drill-down последовательности и систему агрегатов

Аппроксимирующие – алгоритмы, осуществляющие сжатие первоначальных данных для уменьшения объема результирующего куба

Синтаксические методы.

DWARF.

Создатели: Yannis Sismanis, Antonios Deligiannakis, Nick Roussopoulos, Yannis Kotidis

Представлен на VLDB ’02. Последняя работа – VLDB ’04.

Основная идея : Куб «сжимается», за счет удаления избыточности хранимой обычно информации.

Синтаксические методы.

DWARF. Виды Избыточностей

- **Префиксная избыточность –** Пусть у нас есть куб с измерениями a,b и c. Каждое значение измерения a участвует в 4х группировках (a,ab,ac,abc) и, возможно, много раз в каждой из сгруппированных таблиц.
- **Суффиксная избыточность -** Если 2 или более сгруппированные таблицы разделяют одинаковый суффикс (например, abc и bc)

Синтаксические методы.

Свойства DWARF куба.

- Это ациклический ориентированный граф с одной корневой вершиной и имеющий ровно D уровней, где D-число измерений.
- Вершины D уровня (листья) содержат ячейки следующего формата [ключ, агрегирующее значение]
- Вершины на уровнях, отличных от D уровня (не-листовых) содержат ячейки следующего формата: [ключ, указатель]. Ячейка C в не-листовой вершине уровня I указывает на вершину уровня i+1, которую она мажорирует (обобщает). C – родительская вершина для мажорируемой вершины.
- Каждая вершина содержит специальную ячейку, которая содержит псевдо значение ALL как ключ. Эта ячейка содержит или указатель на не-листовую вершину или на агрегирующее значение листовой вершины.
- Ячейки на iом уровне содержат ключи, являющиеся значениями iтого измерения куба. Внутри одной вершины не может встречаться повторения ключа.
- Каждая ячейка Ci на iом уровне структуры, отвечает последовательности из i ключей, входящих в путь от корня до ячейки. Такой путь отвечает оператору group by с (D-i) не указанными измерениями. Все группировки, содержащие в качестве префикса, будут относиться к ячейкам, являющимся потомками Ci в структуре куба. Для всех подобных группировок их общий префикс будет хранится единожды.
- Когда две или более группировки создают одинаковые вершины и ячейки, их хранение обобщается(поглощается), так чтобы хранить только одну копию. В таком случае результирующая вершина будет достижима более чем одним путем из корня, причем все пути будут иметь одинаковый суффикс.

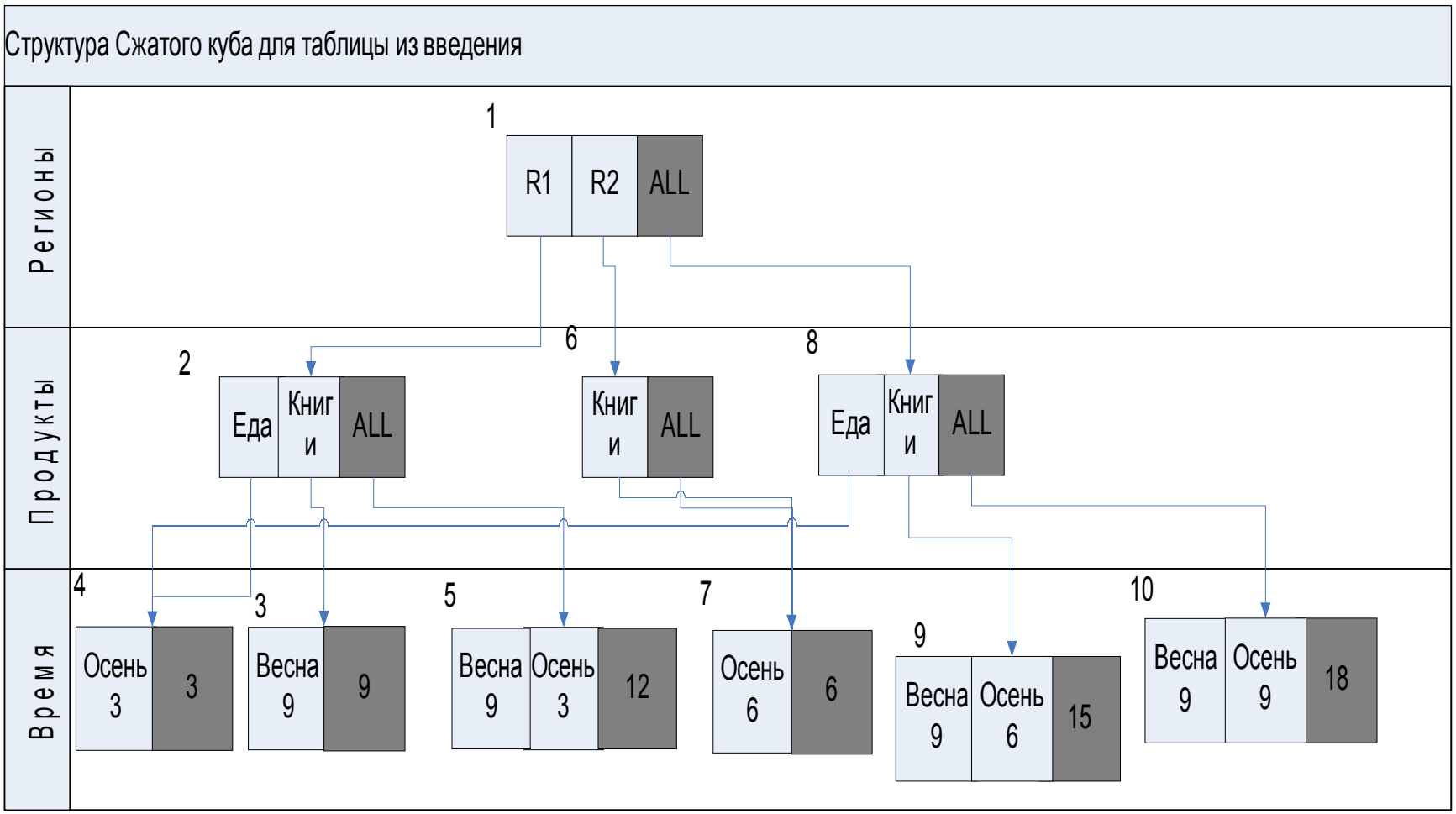
Синтаксические алгоритмы.

DWARF. Алгоритм создания куба

Предполагается, что фактическая таблица лексикографически сортирована

- Для первого кортежа на каждом уровне создаются узлы
- В дальнейшем (из-за сортировки), кортежи будут разделять общий префикс. Отсутствие общего префикса означает необходимость считать агрегаты (добавлять ALL) для данного уровня (узла) (избавляться от суффиксной избыточности).

DWARF. Пример куба.



DWARF. Обработка запросов.

- Точечные запросы выполняются последовательным разыменованием пути в структуре куба. Таким образом, этот вид запросов выполняется намного быстрее, чем в аналогичных алгоритмах или базовых таблицах, за счет того, что каждый запрос требует ровно D обращений к вершинам куба, где D – число измерений (уровней) куба.
- Интервальные запросы включают все хотя бы одно измерение с интервалом значений. Для каждого из ключей iого уровня, попадающего в интервал строятся рекурсивные подзапросы к нижележащим подкубам, что тоже достаточно просто по структуре
- Оптимизаций по выполнению обратных подзапросов этот алгоритм не дает, но возможно его сочетание с какими-либо другими алгоритмами, ориентированными на обратные подзапросы.

DWARF. Выводы.

При использовании этого алгоритма структура куба сжимается синтаксически. Префиксная и суффиксная избыточности устраняются за счет создания лучшей системы адресации и хранения ячеек. Алгоритмы, предложенные для создания и модификации кубов с использованием данной структуры, являются наилучшими из всех синтаксических решений на данный момент.

Оценка сложности алгоритма:

Число ячеек куба $\sim O(T\frac{d^{\log_C T+1}}{\log_C T!}) = O(d\lg T^{1+\frac{1}{\log_d C}})$

Семантические методы.

Quotient Cube.

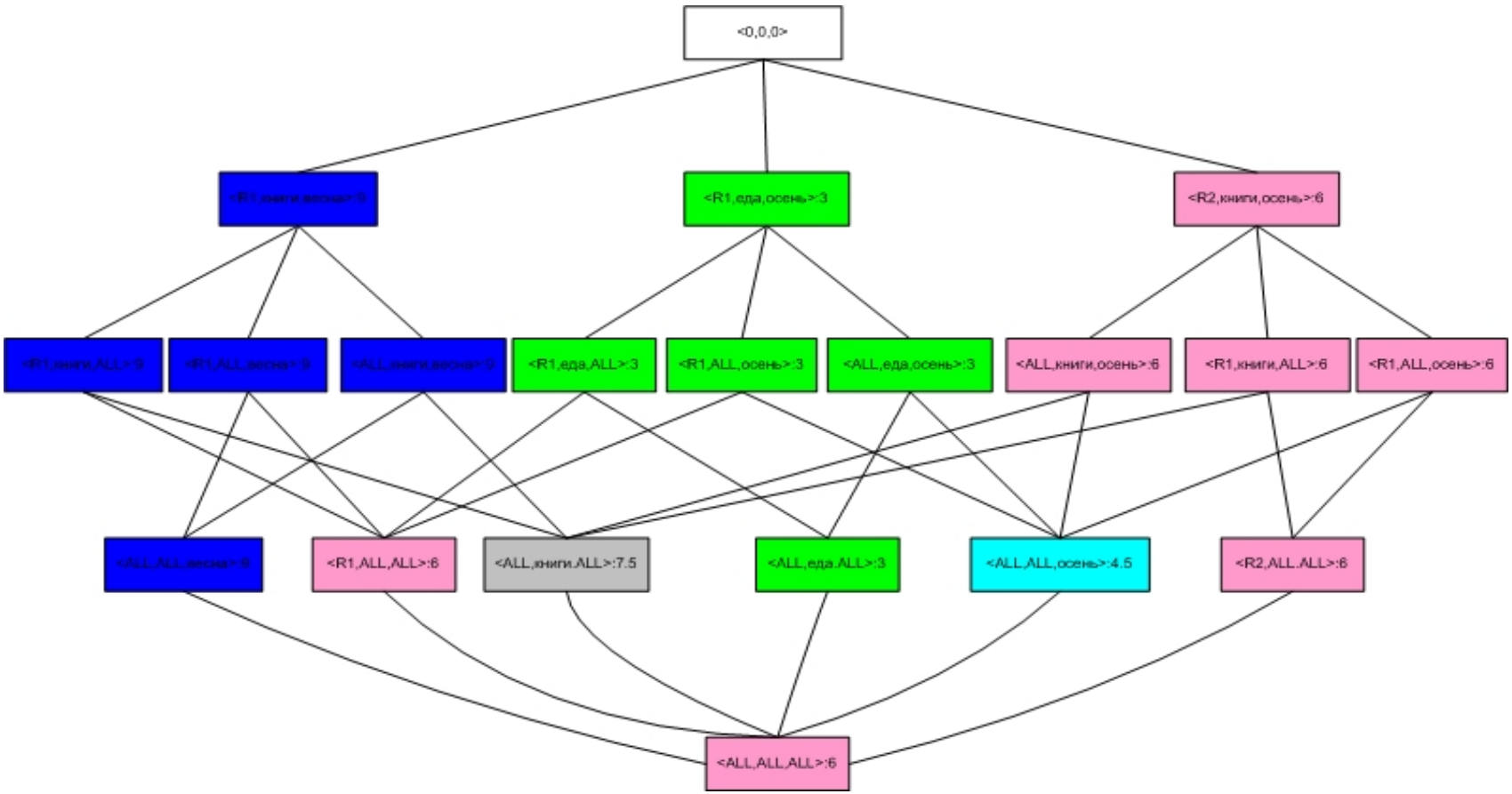
Создатели: Laks V.S., Lakshmanan, Jian Pei, Yan Zhao (Master Degree Work)

Представлен: SIGMOD ’03. Последняя работа – 2005

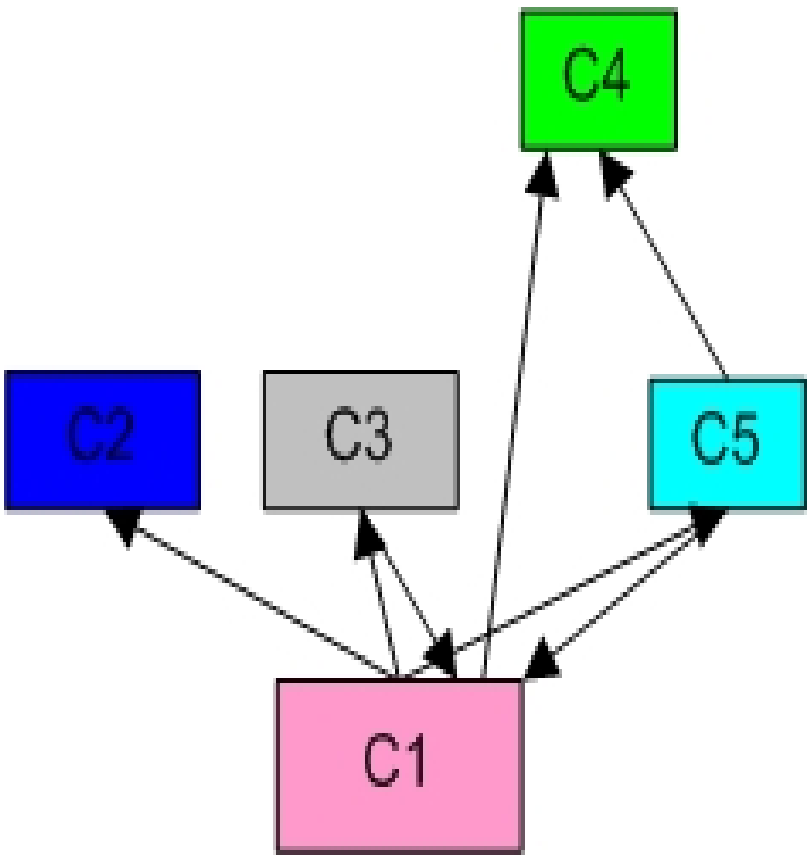
Основная идея: Семантическое сжатие куба, за счет разбиения на классы эквивалентности с сохранением roll-up\drill-down отношений.

Семантические методы.

Quotient Cube. Разбиение по значению функции AVG.



Quotient Cube. Разбиение по значению функции. (продолжение)



Выбор агрегирующей функции в данном случае не важен, т.к. у нас есть возможность двигаться в обе стороны по roll-up\drill-down отношениям. Например, мы идем вверх от ячейки (ALL,ALL,ALL) в C1 в ячейку (ALL,книги,ALL) в C3, а потом в (R2,книги,ALL) в C1. Противоречие.

Quotient Cube. Определения. стр 1

Выпуклое множество

Пусть $\langle P, \leq \rangle$ - частично упорядоченное множество, - выпуклое множество, если

$$\Phi = \{\leq, \geq\}$$

$$\forall x, y \in C \; \exists \{z_j\} \subseteq C, x \Phi z_1 \Phi ... \Phi z_j \Phi y, \; j \in \{N, 0\}$$

Связанное Разбиение.

Пусть f – агрегирующая функция. Тогда мы определим $\equiv_f$ как рефлексивное, транзитивное замыкание отношения **R**.

R: Для a, b в структуре куба R(a,b) выполняется, если:

значение f(a) = f(b)

a>b или b>a

Для монотонных агрегирующих функций (Min, Max, Sum, Count, Top-k) связанное разбиение для куба выпукло и единственно.

Quotient Cube. Определения. стр 2

Покрытие.

Ячейка c **покрывает** базовый кортеж t, если существует roll-up путь от t к c (c<t)

Множеством покрытия ячейки называется множество ячеек, которые она покрывает.

Разбиение по покрытию.

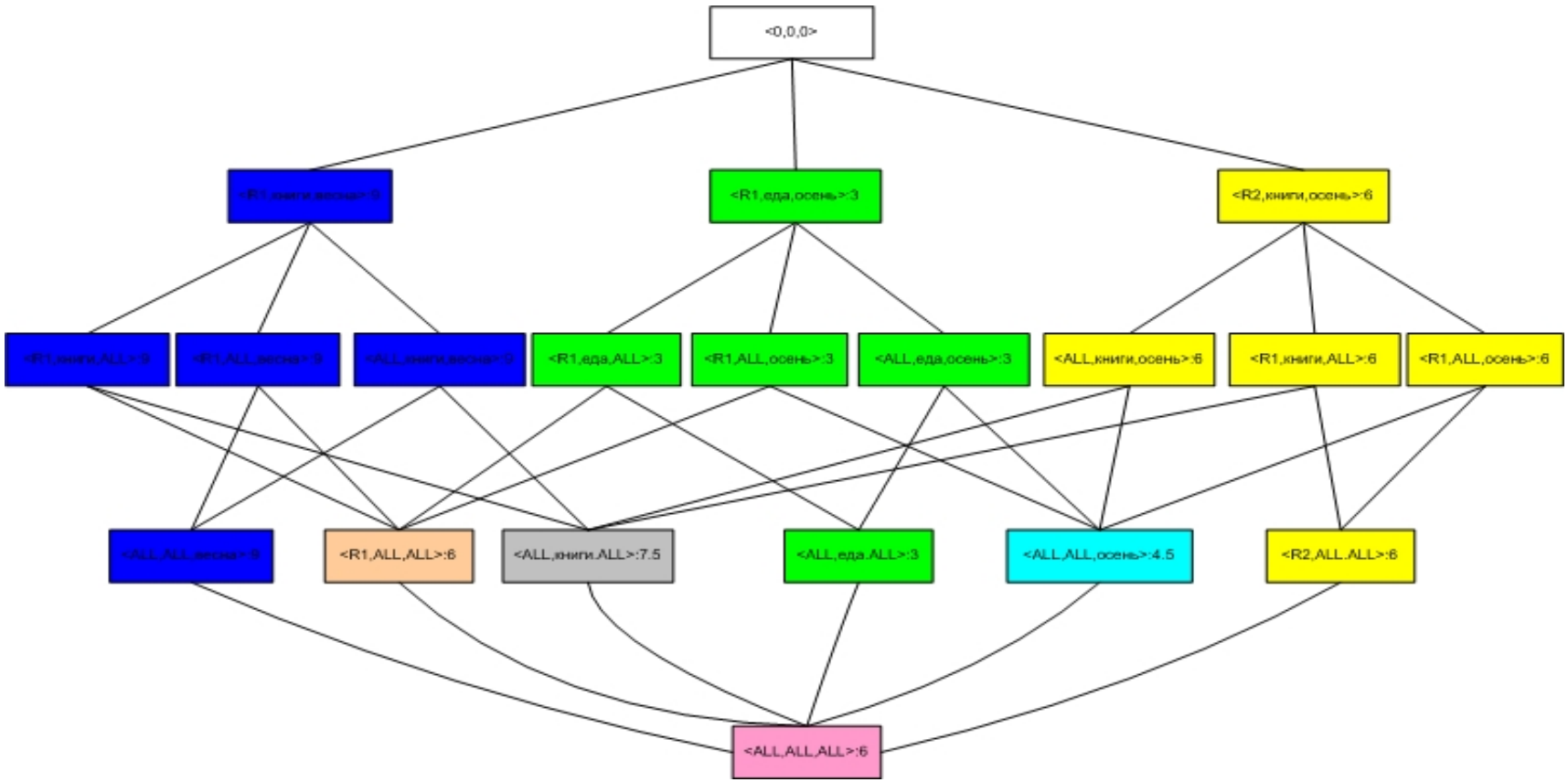
Ячейки a и b. (эквивалентны по покрытию) если их множества покрытия совпадают.

Теорема.

Разбиение по покрытию – связанное. Эквивалентные по покрытию ячейки совпадают по значениям любых агрегирующих функций. У каждого класса в разбиении по покрытию есть уникальная верхняя грань.

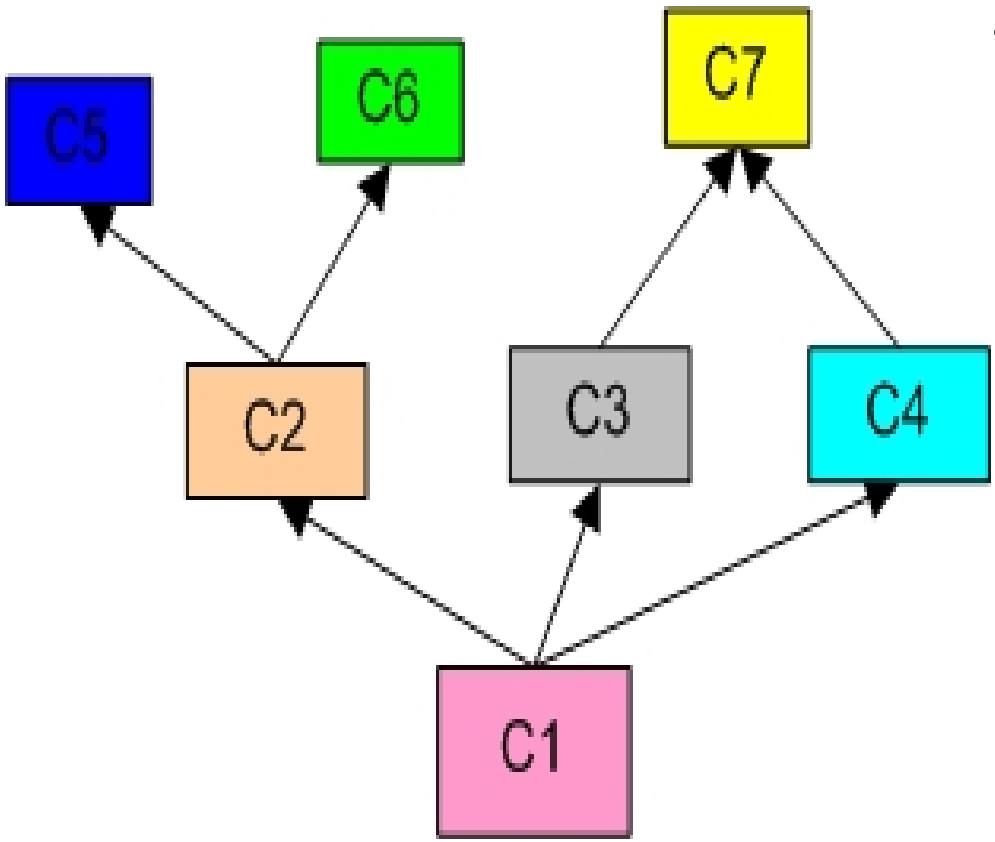
Семантические методы.

Quotient Cube. Разбиение по покрытию.



Семантические методы.

Quotient Cube. Разбиение по покрытию.(продолжение)



Таким образом, для получившегося куба нам нужно хранить только две ячейки на класс, верхнюю и нижнюю грань. Остальные ячейки выводятся из верхней и нижней грани, т.к. классы разбиения по покрытиям «полны» в смысле введенного частичного порядка.

Семантические методы.

Quotient Cube. Реализация.

Предложено две структуры хранения данных для Quotient Cube: QC – Tables и QC-Trees.

- QC-Table – простая таблица, в которой для каждого класса хранится верхняя и нижняя грань. Предложены алгоритмы создания QC-Table и вставки\удаления классов. Такие таблицы неэффективны по времени обработки запросов, т.к. каждый запрос требует просмотра почти всей таблицы.

Семантические методы.

Quotient Cube. QC-Trees. Основное.

Определение: дерево с разделением префиксов, где грани представляются строчками, а связи отражают необходимые drill-down отношения

Основной идеей создания QC-деревьев является тот факт, что верхние грани классов куба хранят всю необходимую информацию для выполнения запросов. Т.е. при данном подходе не хранятся даже нижние грани классов. При такой структуре хранения данных Quotient кубы по эффективности выполнения запросов опережают DWARF кубы. Достигается это превосходство за счет такой аналогичного по смыслу устранения суффиксной и префиксной избыточностей. QC-деревья позволяют устранять суффиксную избыточность за счет того что хранятся только классы, а не отдельные ячейки. Префиксная избыточность устраняется за счет свойств самой структуры QC-tree.

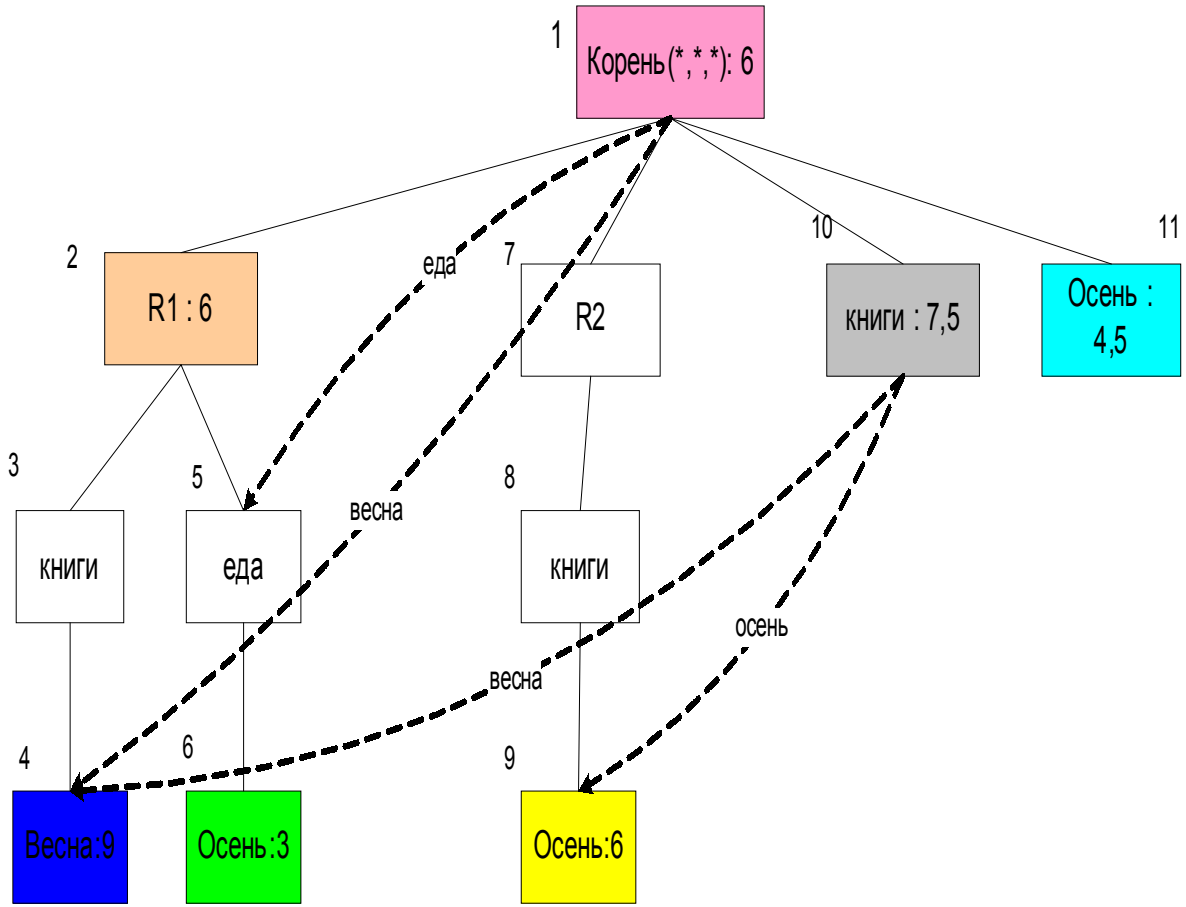
Quotient Cube. QC-Trees. Определение.

QC-Tree для Quotient куба Q – это орграф (V,E) в котором:

- $E = E' \cup E''$,состоящий из ребер E' и связей E'', таким образом, что (V,E') –дерево
- Каждая вершина содержит метку, равную значению одного из измерений
- Для каждой верхней грани b, существует и единственна вершина $v \in V$ такая, что строчное представление b совпадает с последовательностью меток вершин на пути от корня к v в (V,E'). Эта вершина хранит агрегирующее значение для b.
- Предположим D и C - классы в Q, $b_1 = (x_1, K, x_n)$ è $b_2 = (y_1, K, y_n)$ - их верхние грани и, что C – потомок D (например, что C напрямую специализирует (drills-down) D) в Q. Тогда для каждого измерения D_i, на котором различаются b1 и b2, существует либо грань дерево, либо связь (маркированная yi) из вершины $\langle x_1, K, x_{i-1} \rangle$ в вершину $\langle y_1, K, y_i \rangle$

Семантические методы.

Quotient Cube. QC-Trees. Пример.



Класс	Верхняя грань
C1	(ALL,ALL,ALL)
C2	(R1,ALL,ALL)
C3	(ALL,книги,ALL)
C4	(ALL,ALL,осень)
C5	(R1,книги,весна)
C6	(R2,еда,осень)
C7	(R2,книги,осень)

Quotient Cube. Выполнение запросов. Точечные запросы (Point Queries).

Пусть c (*,*,…,v1,*,…,*,v2,*,…,vk,*,…,*), где vi!=*, значения измерений Li, i=1,…,k.

Алгоритм.

1 Curr_node = корень; curr_val=v1

2 На любом узле curr_node искать дугу с меткой curr_val,

если существует:curr_node = потомок по найденной дуге; curr_val = vi+1

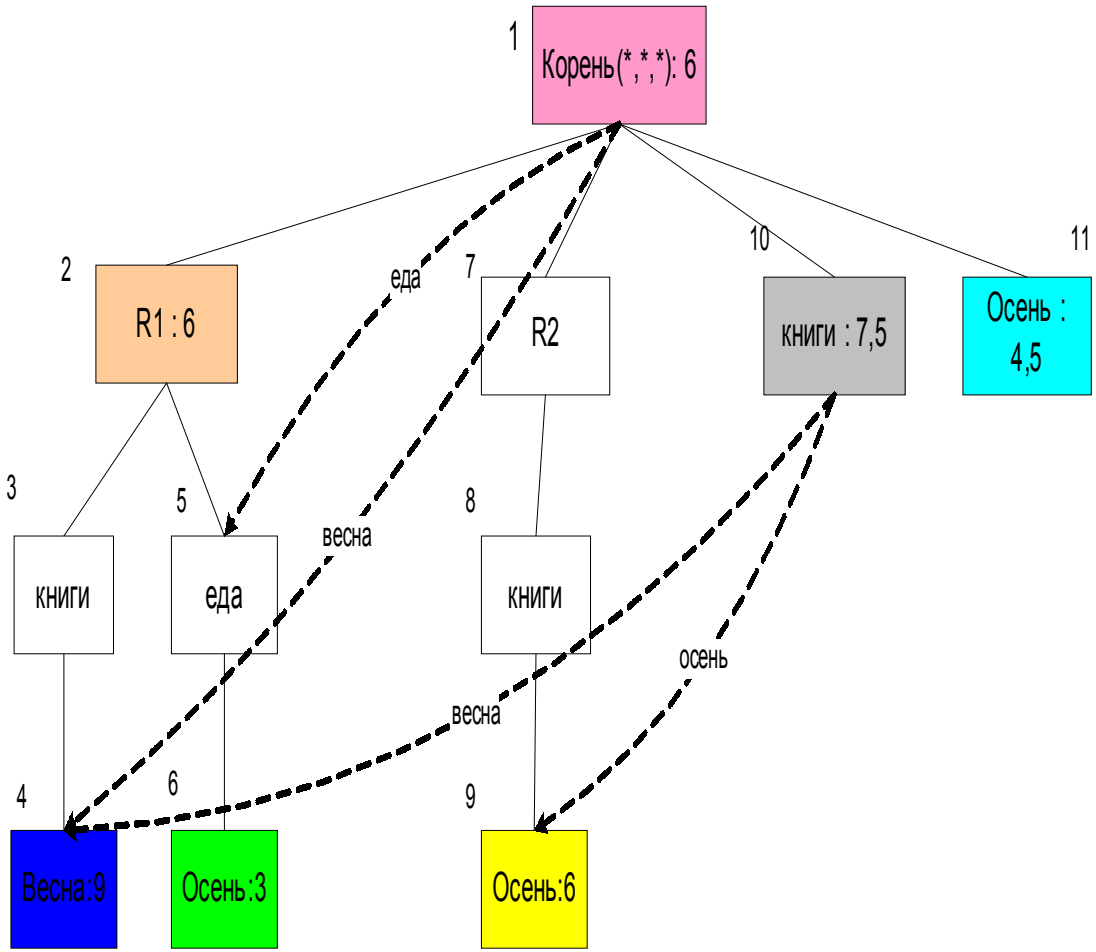
иначе: проверить последнее измерение j, по которому у curr_node есть потомок.

Если j>=Li, тогда c в кубе не появится.

Иначе: curr_node = потомок по измерению j, снова повторяем 2.

Семантические методы.

Quotient Cube. Выполнение запросов. Точечные запросы (Point Queries).



Примеры

- (R2,*,осень) – начинаем с корня, находим вершину 7, в вершине 7 ищем «осень», берем потомка по измерению, продукты, попадаем в 9 – есть ответ.
- (R2,*,весна) – все тоже самое но в 9 мы будем пытаться найти «весна» - такой ячейки нет
- (*,еда,*) – в 5, но там нет значения, проваливаемся в 6 – ответ

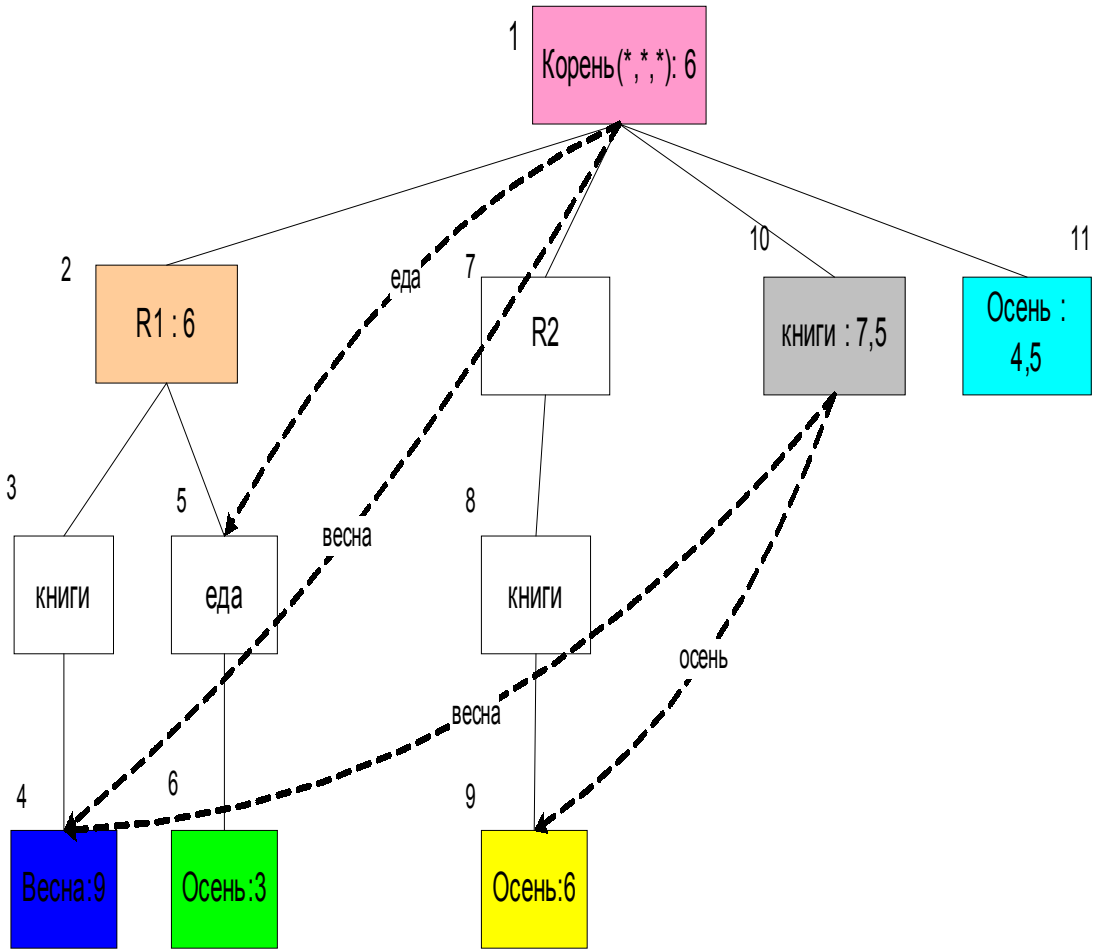
Семантические методы.

Quotient Cube. Выполнение запросов. Интервальные запросы (Range Queries).

- Запрос:
  (*,…,v1,*,…,{up1,..,upr},…,vi,*,…,{wq1,...,wqs},*…vk,*…)
- Можно превратить такой запрос в серию точечных запросов, но эффективней во время обработки каждого интервала исключать все недостижимые ячейки

Семантические методы.

Quotient Cube. Выполнение запросов. Интервальные запросы (Range Queries).



Примеры:

(*,{книги,еда},весна)

порядок обработки следующий:

- корень->книги->весна – возвращается значение
- корень -> еда - но не можем попасть в «весну»

Семантические методы.

Quotient Cube. Выполнение запросов. Обратные запросы (Iceberg Queries).

Прямой оптимизации по выполнению обратных запросов метод не дает. Но возможно построение индексов B+ деревьев по мерам в QC-деревьях. Это поможет в выполнении обратных запросов, не накладывающих ограничений на измерения.

В случае наличия ограничений в запросе, можно оставить в QC-дереве только вершины, удовлетворяющие условию и их предков. Получится поддерево, на котором нужно выполнить интервальный запрос.

Семантические методы.

Quotient Cube. Выводы.

- Доказана теорема, о том, что QC-дерево для каждого Quotient куба уникально. Существуют алгоритмы создания\поддержки QC-деревьев. Выполнение запросов на QC-деревьях сильно отличается от аналогичных алгоритмов в других методах, т.к. необходимо «выводить» содержимое ячейки из верхней и нижней гранях класса. Аналогично, поддержка изменений информации требует пересмотра структуры классов, что, в общем случае, замедляет внесение изменений.
- Метод Quotient кубов дает быстрый ответ на так называемые «умные» roll-up запросы (Intelligent Roll-Up Queries), которые формулируются следующим образом: «Каковы наиболее общие условия, при которых значение меры является следующим?». Ответ на данные запросы дается во время построения Quotient кубов, т.к. это и есть верхние и нижние границы кубов.

Аппроксимирующие методы.

Wavelets in OLAP.

Создатели:Jerey Scott Vitter,Min Wangy,Bala Iyer

Представлен: CIKM ‘98

Основными посылками явились 2 наблюдения.

- Часто встречаются ситуации, когда пользователь предпочтет не точный результат, а некоторый приближенный, особенно если последний он может получить на порядок быстрее
- Для ряда приложений возможны ситуации, когда недоступен источник данных, а существует необходимость обработать запрос, даже выдав не совсем точный результат

Отметим, что метод нацелен на оптимизацию только одного типа запросов – интервальных запросов (range-sum query).

Аппроксимирующие методы.

Wavelets in OLAP. Описание метода

- Сначала формируется куб частичных сумм (partial sum cube) – куб, в котором ячейки представляют собой многомерные суммы всех прилежащих ячеек. Это представление данных хорошо тем, что ячейки куба частичных сумм монотонно не убывают по всем измерениям, по этому аппроксимация этого куба является более точной, нежели аппроксимация исходного. Для ответа на интервальный запрос требуется лишь оценка значения в одной ячейке, а не целого набора ячеек.
- После создания куба частичных сумм, создается новый куб, в котором каждая ячейка представляет собой натуральный логарифм соответствующей ячейки из куба частичных сумм. Таким образом, еще больше «сглаживаются» колебания значений.
- Потом формируется декомпозиция получившегося куба, основанная на вейвлетах (хоаровских вейвлетах), - декомпозиция неоднократно применяется для каждого из измерений, таким образом, куб «складывается» по каждому из измерений.
- На следующем этапе мы выбираем C главных коэффициентов вейвлетовой декомпозиции данных. Остальные N-C коэффициентов мы получаем, усредняя начальные C. Таким образом, можно сильно сократить объем хранимых данных, почти не теряя в точности результата. Скорость обработки запросов растет, соответственно с уменьшением объема хранимых данных и возможностью обработки запросов обращением только к одной ячейке и последующими преобразованиям данных. По скорости обработки интервальных запросов этот алгоритм, при соответствующем выборе коэффициентов, превосходит все существующие алгоритмы при малом значении ошибки данных.

Wavelets in OLAP. Постановка задачи.

- Range-Sum Query

$$Sum(l_1 : h_1, \ldots, l_d : h_d) = \sum_{l_1 \le i_1 \le h_1} \ldots \sum_{l_d \le i_d \le h_d} A[i_1, \ldots, i_d]$$

- Определение Partial-Sum Cube.

$$P[x_1, \ldots, x_d] = Sum(0 : x_1, \ldots, 0 : x_d) = \sum_{0 \le i_1 \le x_1} \ldots \sum_{0 \le i_d \le x_d} A[i_1, \ldots, i_d]$$

Wavelets in OLAP. Декомпозиция.

Положим, у нас есть набор одномерный значений: [2,2,7,11]

Преобразование:

[5.5,3.5,0,2]

Разрешение	Среднее значение	Детальные коэффициенты
4	[2,2,7,11]	
2	[2,9]	[0,2]
1	[5.5]	[3.5]

Модель CPM. Описание.

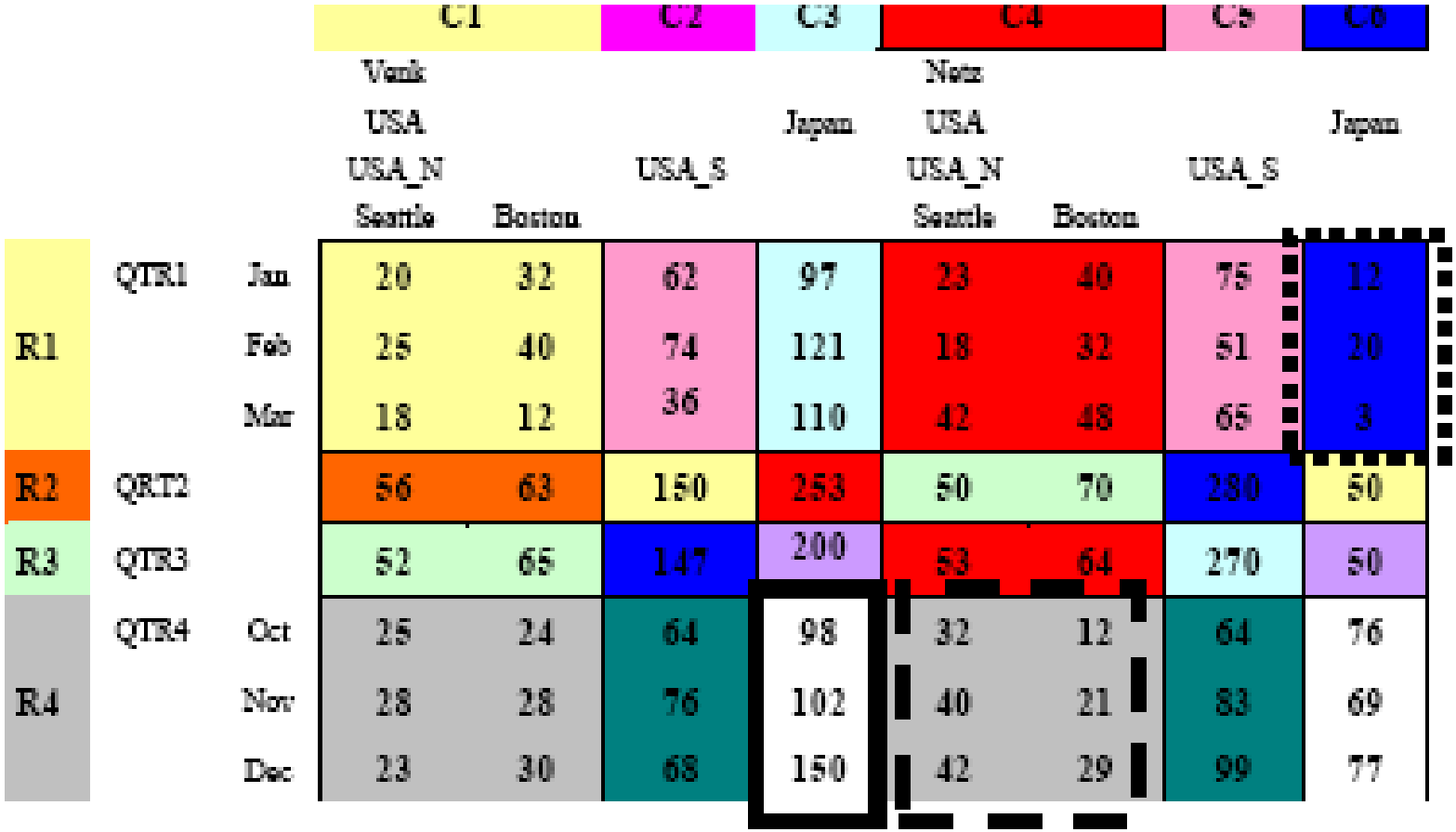
- Создатели:

Andreas S. Maniatis,Panos Vassiliadis, Spiros Skiadopoulos, Yannis Vassiliou

- Идея: Систематизация подхода к визуализации OLAP-данных. Создание модели CPM (Cube Presentation Model)

Визуализация OLAP-данных.

Модель CPM. Пример.



			C1		C2	C3	C4		C5	C6
			Vank				Netz			
			USA			Japan	USA			Japan
			USA_N		USA_S		USA_N		USA_S	
			Seattle	Boston			Seattle	Boston		
	QTR1	Jan	20	32	62	97	23	40	75	12
R1		Feb	25	40	74	121	18	32	51	20
		Mar	18	12	36	110	42	48	65	3
R2	QRT2		56	63	150	253	50	70	280	50
R3	QTR3		52	65	147	200	53	64	270	50
	QTR4	Oct	25	24	64	98	32	12	64	76
R4		Nov	28	28	76	102	40	21	83	69
		Dec	23	30	68	150	42	29	99	77

Выводы и направления возможных исследований

- Обратные запросы
- Инкрементальное добавление данных
- Аппроксимирующие методы
- Кубы над потоками данных
- Cube Groups
- Использование разных функций для агрегации по разным измерениям