

数据仓库与数据挖掘

Data warehouse and data mining

丁钰

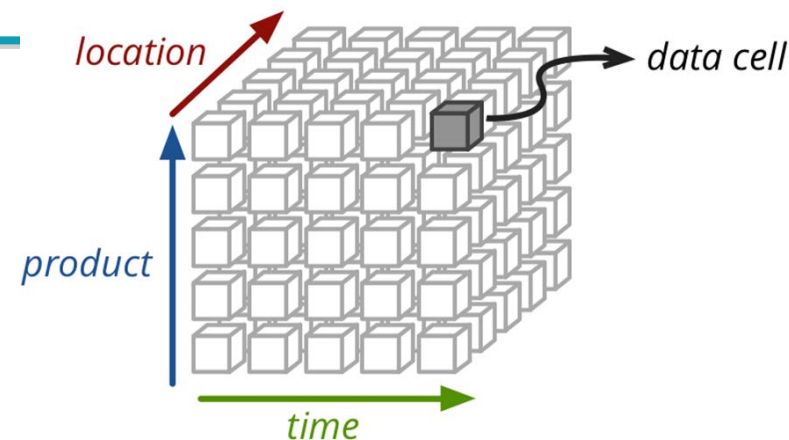
yuding@live.com

南京农业大学人工智能学院

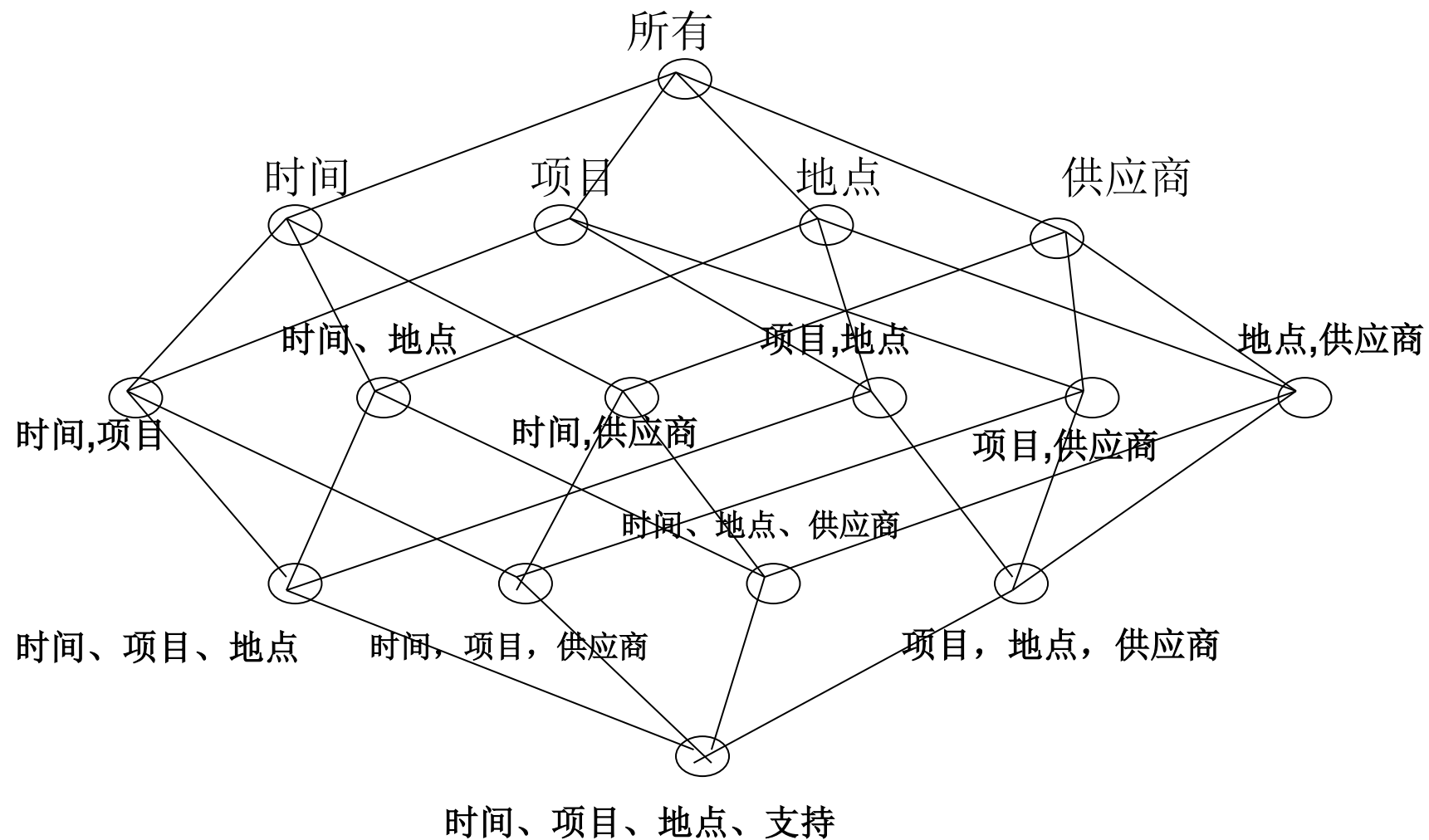
第五章: 数据立方体技术

第5章：数据立方体技术

- 数据立方体的计算：基本概念
- 数据立方体的计算方法
- 数据立方体空间的多维数据分析
- 摘要



数据立方体：一个立方体的格子



0-D (顶点) 长方体

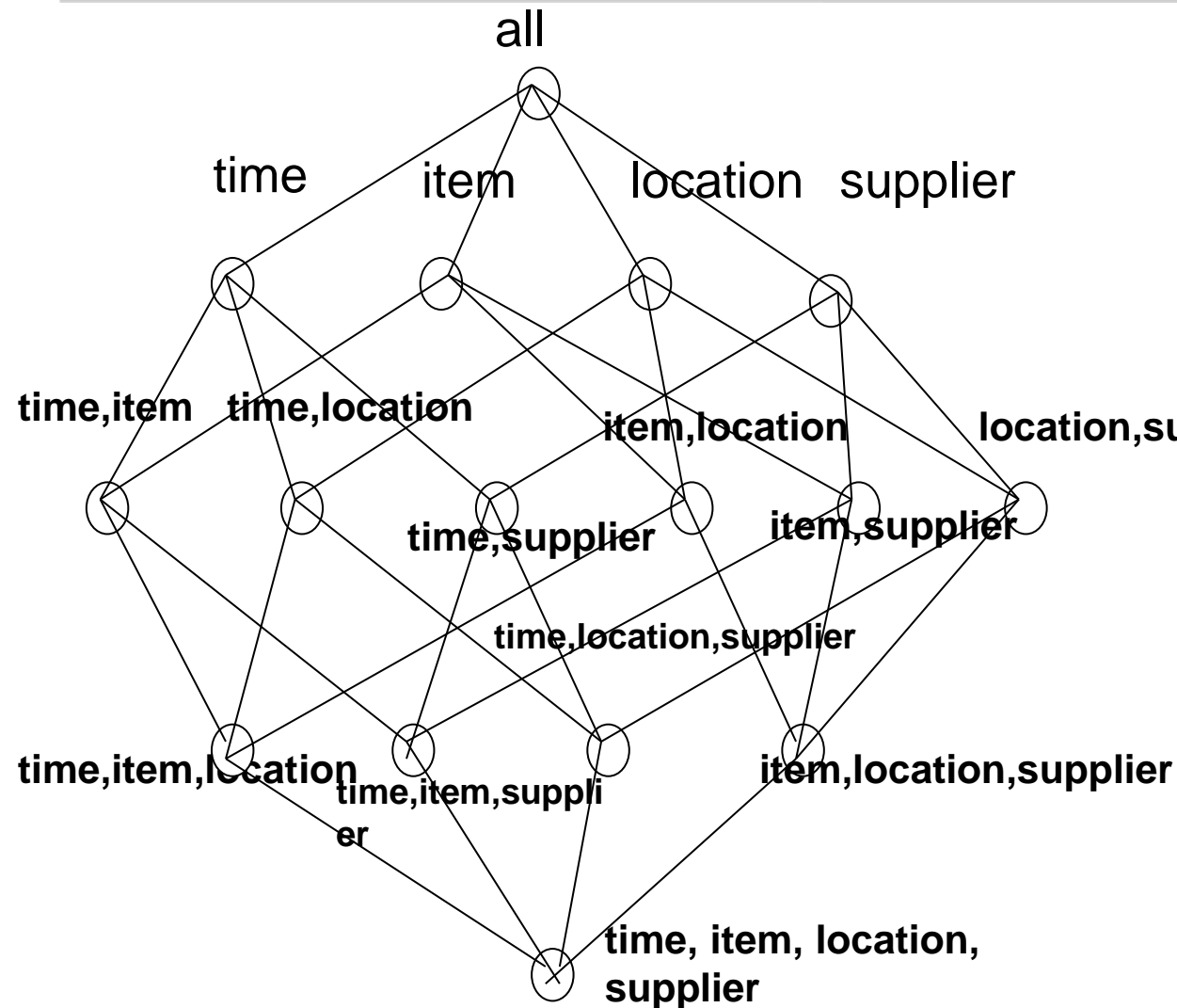
1-D立方体

2-D立方体

3-D立方体

4-D(基)立方体

数据立方体。一个立方体的格子



- 0-D(apex) cuboid ■ Base vs. aggregate cells
- Ancestor vs. descendant cells
- 1-D cuboids ■ Parent vs. child cells
- 2-D cuboids ■ $(*, *, *)$
- $(*, \text{milk}, *, *)$
- $(*, \text{milk}, \text{Urbana}, *)$
- 3-D cuboids ■ $(*, \text{milk}, \text{Chicago}, *)$
- $(9/15, \text{milk}, \text{Urbana}, *)$
- $(9/15, \text{milk}, \text{Urbana}, \text{Dairy_land})$
- 4-D(base) cuboid

立方体物化：完全立方体与冰山立方体

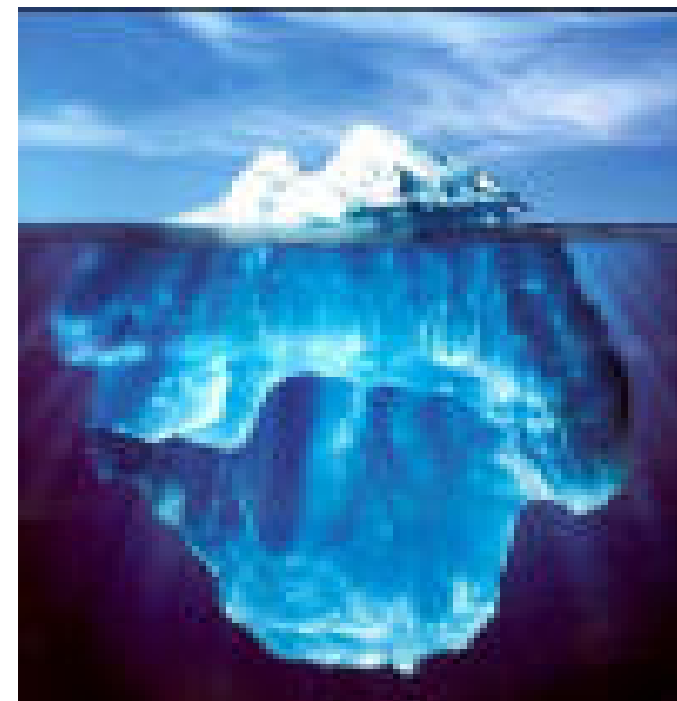
- 完整的立方体与冰山立方体

```
compute cube sales iceberg as  
select month, city, customer group, count(*)  
from salesInfo  
cube by month, city, customer group  
having count(*) >= min support
```



**iceberg
condition**

- 只计算测量值满足冰山条件的单元格
- 在一个稀疏的立方体中，只有一小部分单元格可能是 "在水面上"。
- 例：只显示那些计数不低于100的单元格



为什么是冰山一角？


❑ 计算冰山立方体的优势

- ❑ 不需要保存也不需要显示那些数值低于阈值的单元格（冰山条件）。
- ❑ 高效的方法可以避免计算不需要的、中间的单元格
- ❑ 避免爆炸性增长

❑ 例子。 一个有100个维度的立方体

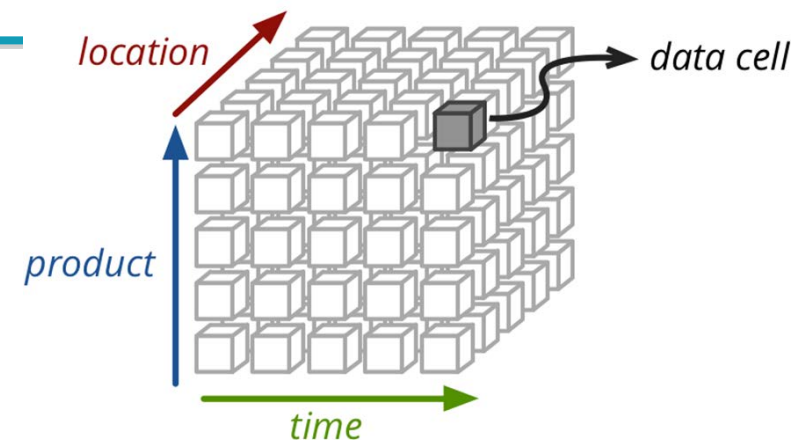
- ❑ 假设它只包含2个基础单元。 $\{(a_1, a_2, a_3, \dots, a_{100}), (a_1, a_2, b_3, \dots, b_{100})\}$
- ❑ 如果 "have count ≥ 1 ", 有多少个聚合单元格？
 - ❑ 答案是。 $(2^{101} - 2) - 4$ (为什么？)
- ❑ 那么，冰山单元（即条件： " have count ≥ 2 " ）呢？
 - ❑ 答案是。 4 (为什么?!)

冰山立方体足够好吗？ 闭立方体和立方体外壳

- 假设立方体P只有2个基础单元。 $\{(a_1, a_2, a_3 \dots, a_{100}):10, (a_1, a_2, b_3, \dots, b_{100}):10\}$ 。
 - 如果 "have count(*) ≥ 10 ", 冰山立方体将包含多少个单元格？
 - 答案： $2^{101} - 4$ （还是太大了！）。
 - 闭立方体。
 - 如果不存在任何单元格d，使d是c的后代，且d与c有相同的度量值，则单元格c是**封闭的**。
 - 例子。同样的立方体P只有3个封闭单元。
 - $\{(a_1, a_2, *, \dots, *): 20, (a_1, a_2, a_3 \dots, a_{100}): 10, (a_1, a_2, b_3, \dots, b_{100}): 10\}$ 。
 - 一个**封闭的立方体**是一个只由封闭单元组成的立方体。
 - 立方体外壳。只涉及少量维度的立方体，例如，2
 - 只计算立方体外壳，其他的尺寸组合可以即时计算。
-  问：对于 $(A_1, A_2, \dots, A_{100})$ ，要计算多少个组合？

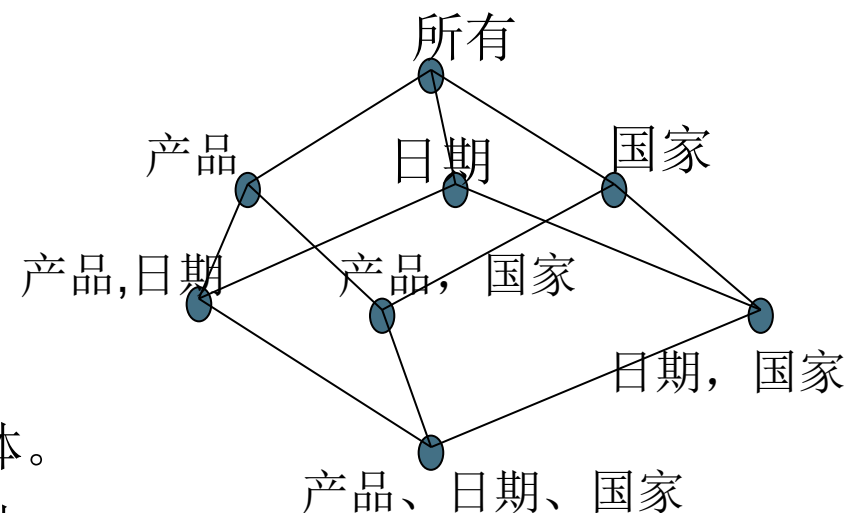
第5章：数据立方体技术

- 数据立方体的计算：基本概念
- 数据立方体的计算方法
- 数据立方体空间的多维数据分析
- 摘要



高效的数据立方体计算：一般策略

- 对维度属性进行排序、散列和分组操作，以便对相关图元进行重新排序和分组。
- 汇总可以从以前计算的汇总中计算，而不是从基本事实表中计算。
 - **最小的子女**：从最小的、以前计算过的立方体计算出一个立方体。
 - **缓存结果**：缓存一个立方体的结果，从该立方体计算其他立方体以减少磁盘I/O。
 - **平摊扫描**：在同一时间计算尽可能多的立方体，以摊销磁盘读取量。
 - **共享排序**：当使用基于排序的方法时，共享跨多个立方体的排序成本
 - **共享分区**：当使用基于哈希的算法时，在多个立方体之间共享分区成本。



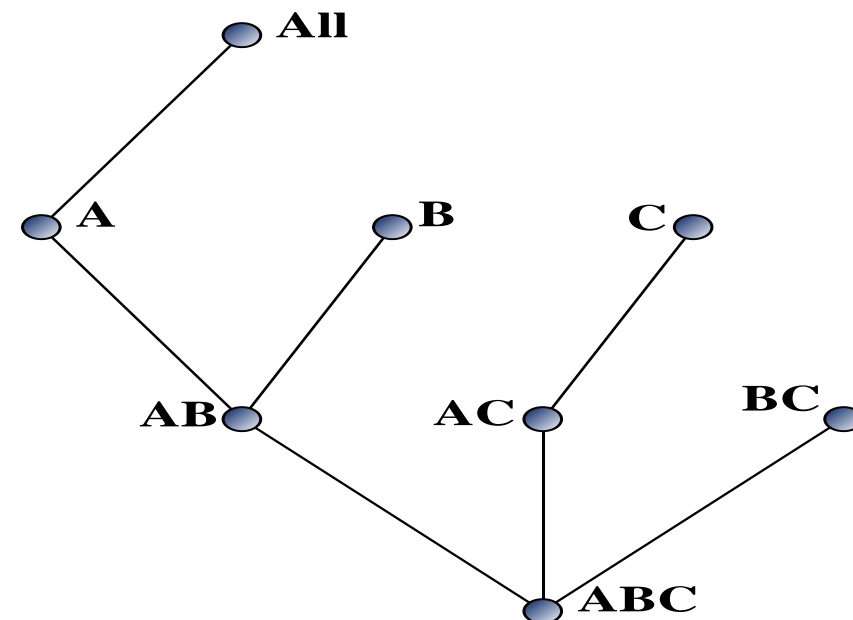
S. Agarwal, R. Agrawal, P. M. Deshpande, A. Gupta, J. F. Naughton, R. Ramakrishnan, S. Sarawagi. On the computation of multidimensional aggregates VLDB'96

多路数组聚合

- 基于数组的 "自下而上" 算法（从ABC到AB， ...）。
- 使用多维块状物
- 同时在多个维度上进行汇总
- 中间聚合值被重新用于计算祖先立方体
- 不能进行 *Apriori* 修剪：没有冰山优化

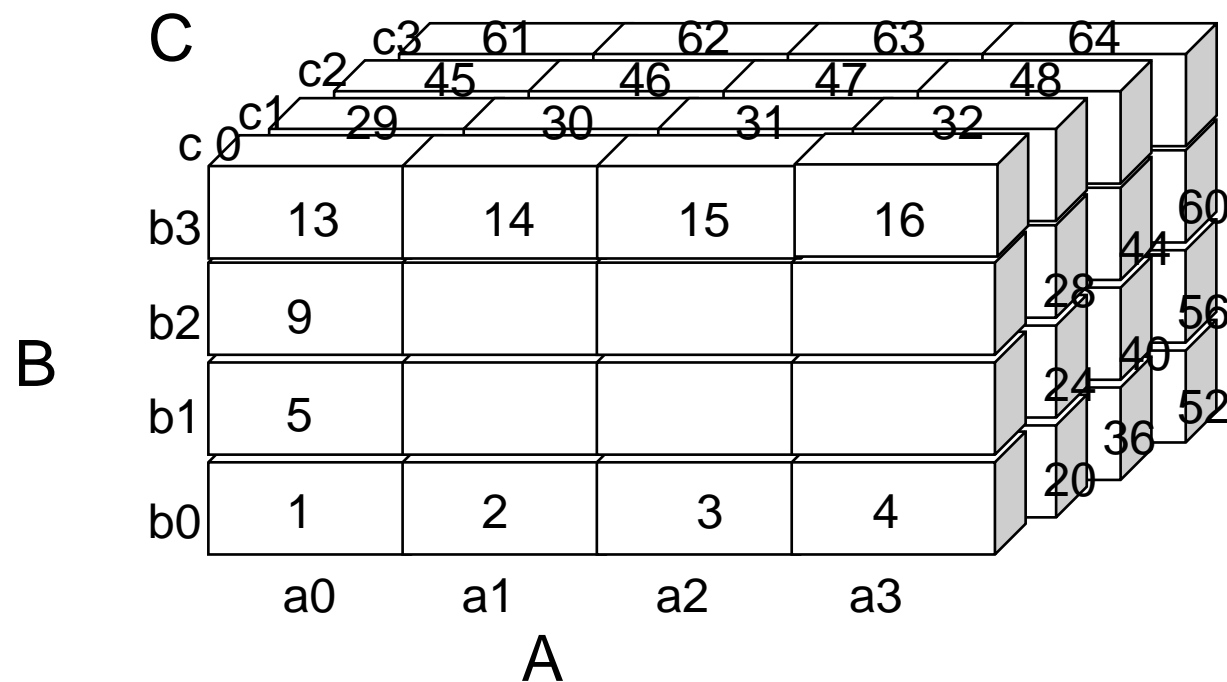
□ 对该方法的评论

- 对少量维度的完全立方体进行高效计算
- 如果有大量的维度，应该使用 "自上而下" 的计算和冰山立方体的计算方法（如BUC）。



立体计算：多路数组聚合(MOLAP)

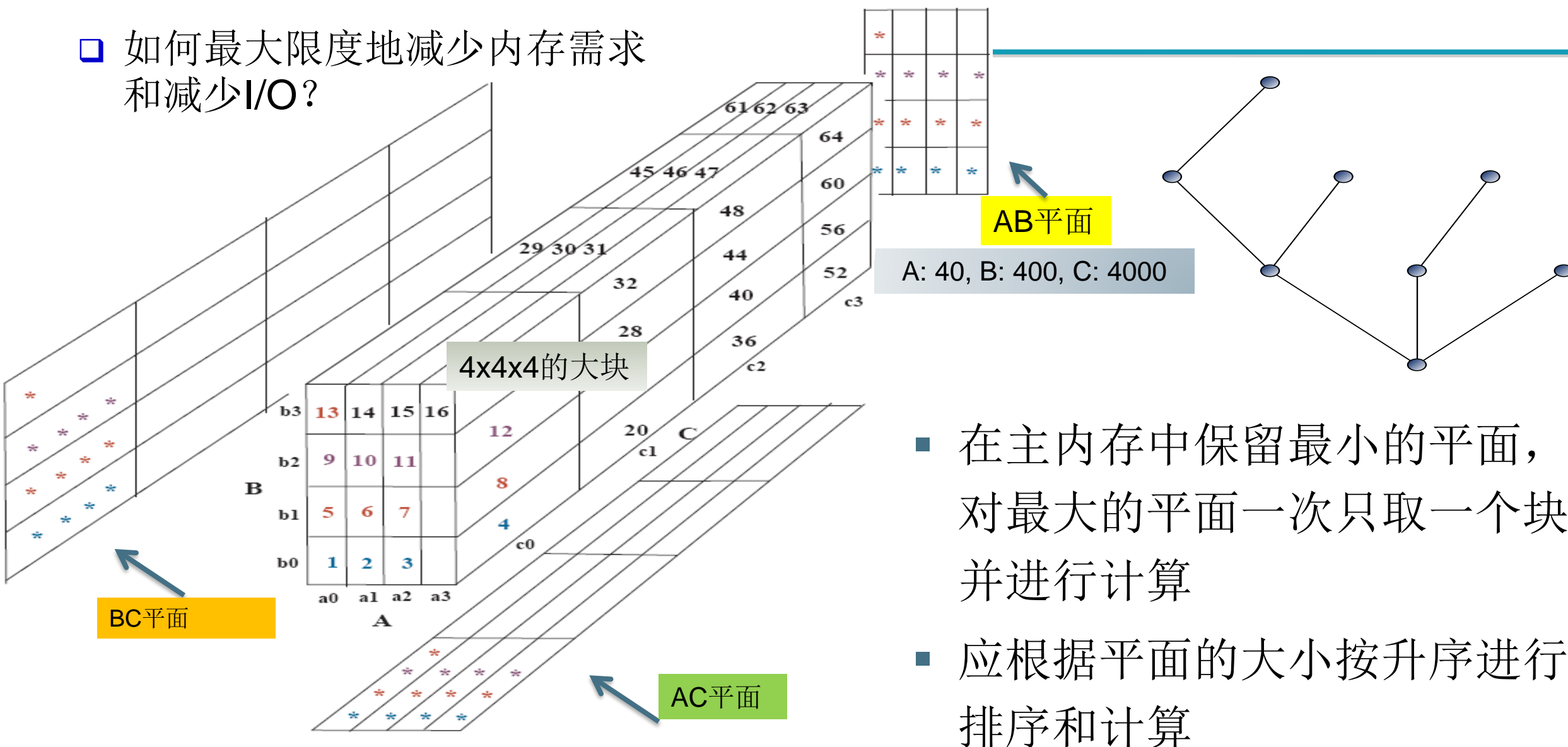
- 将数组分割成小块（适合内存的小子方块）。
- 压缩的稀疏数组寻址。(chunk_id, offset)
- 通过访问立方体单元的顺序，以 "多路 "方式计算总量，使访问每个单元的次数最少，并减少内存访问和存储成本。



做多向聚合的最佳遍历顺序是什么？

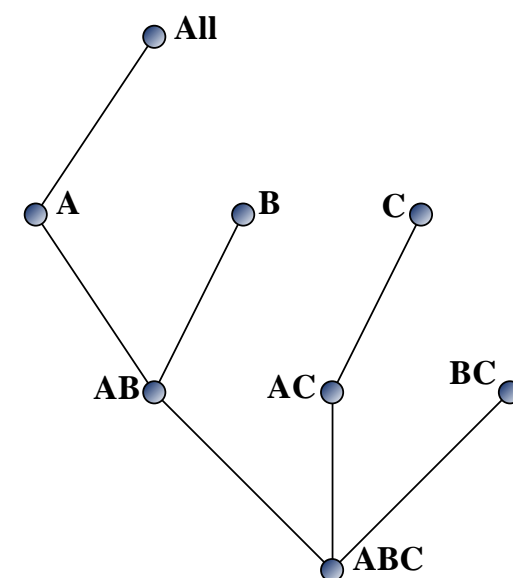
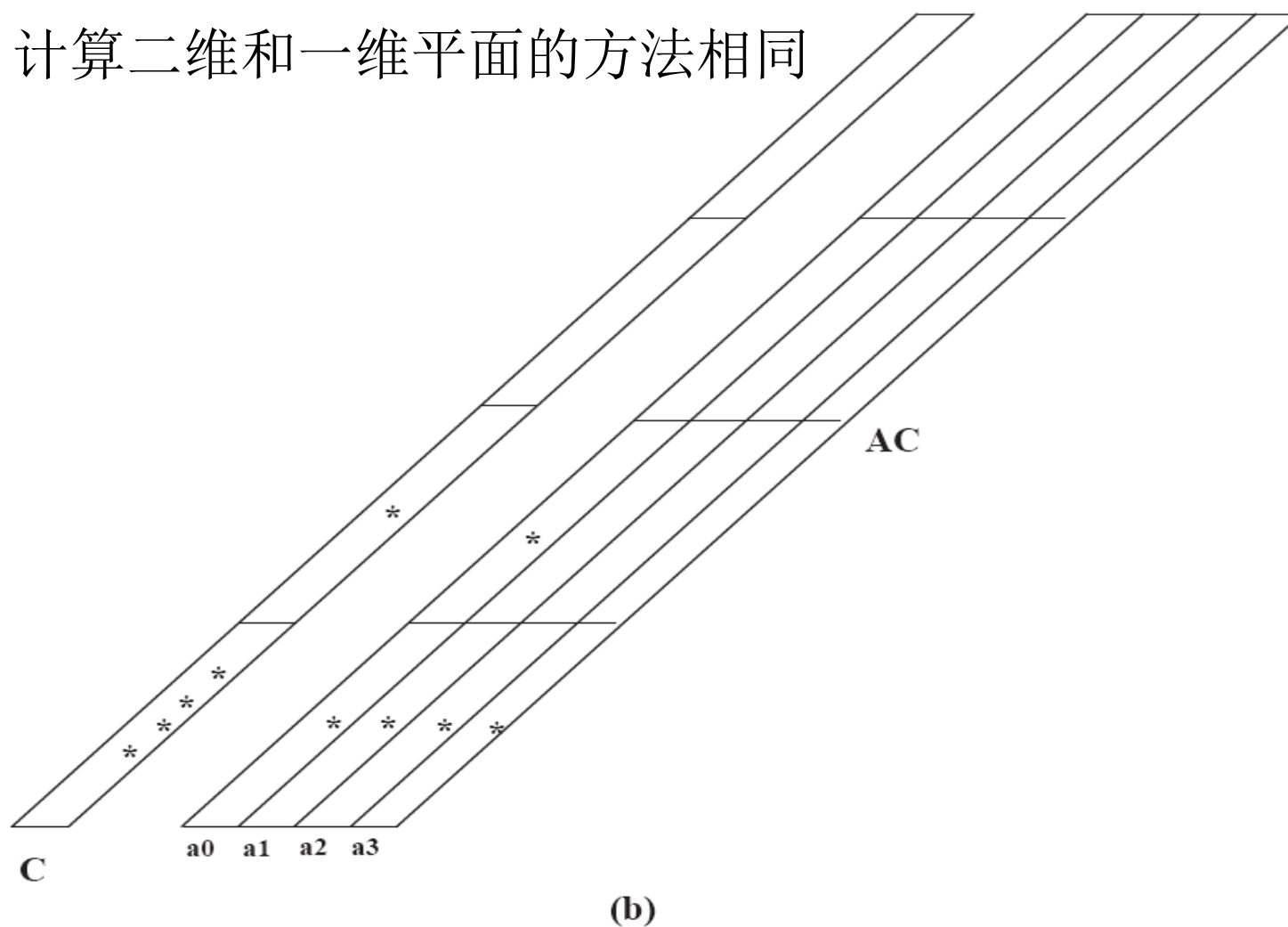
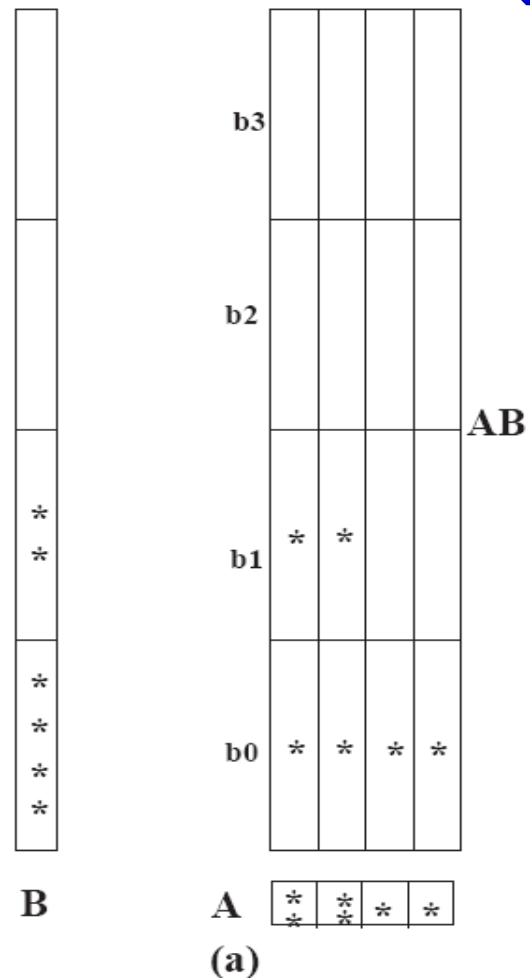
多路数组聚合（3-D到2-D）。

- 如何最大限度地减少内存需求和减少I/O？



多路数组聚合（2-D到1-D）。

□ 计算二维和一维平面的方法相同



立方体计算：逆序计算

- BUC (Beyer & Ramakrishnan, SIGMOD'99)

BUC: 自下而上（立方体） Bottom-Up (cube)

Computation 计算的首字母缩写

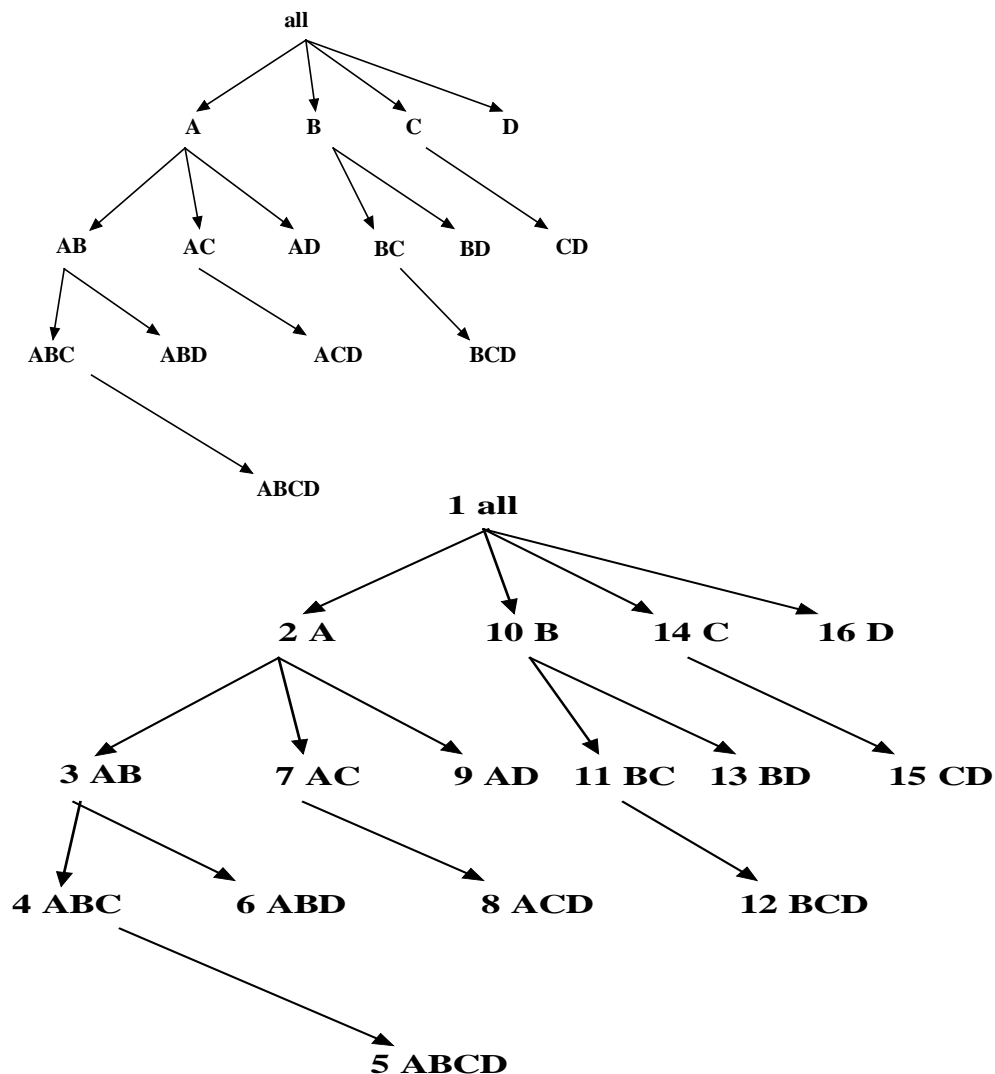
(注意：在我们看来，这是 "自上而下" 的，因为我们把Apex立方体放在上面！)

- 将维度划分为不同的分区，便于冰山修剪

- 如果一个分区不满足 min_sup ，它的后代可以被修剪掉

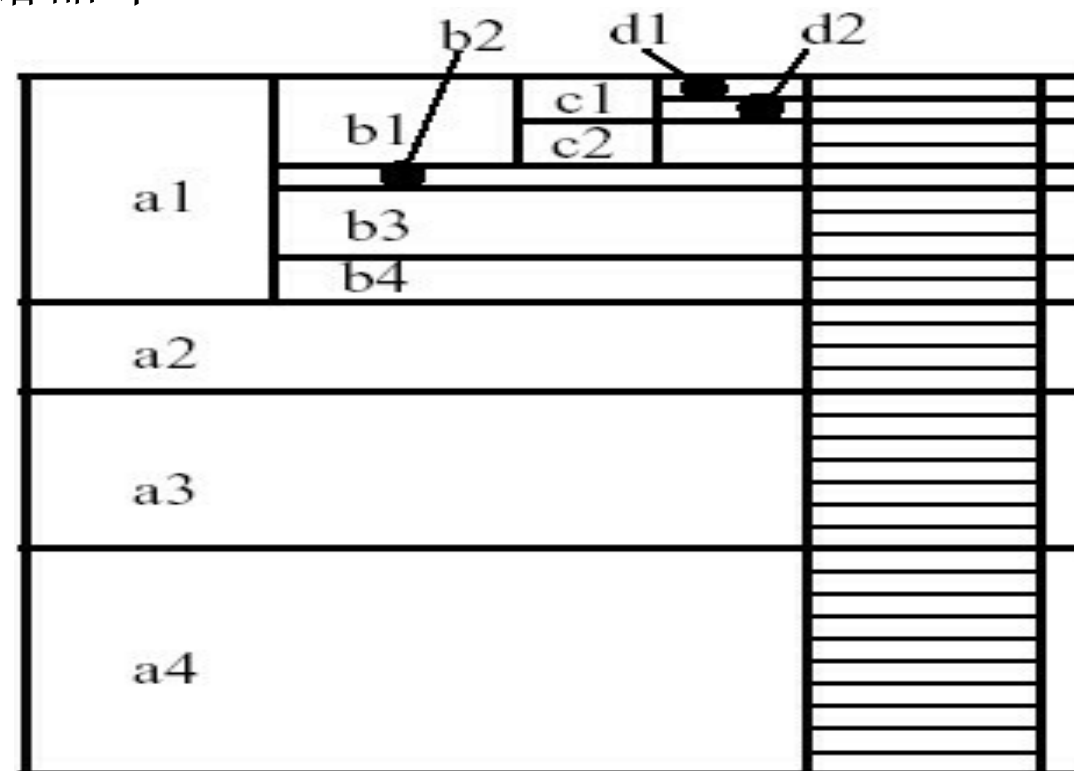
- 如果 $minsup = 1$ 计算完整的CUBE!

- 没有同时进行的聚合



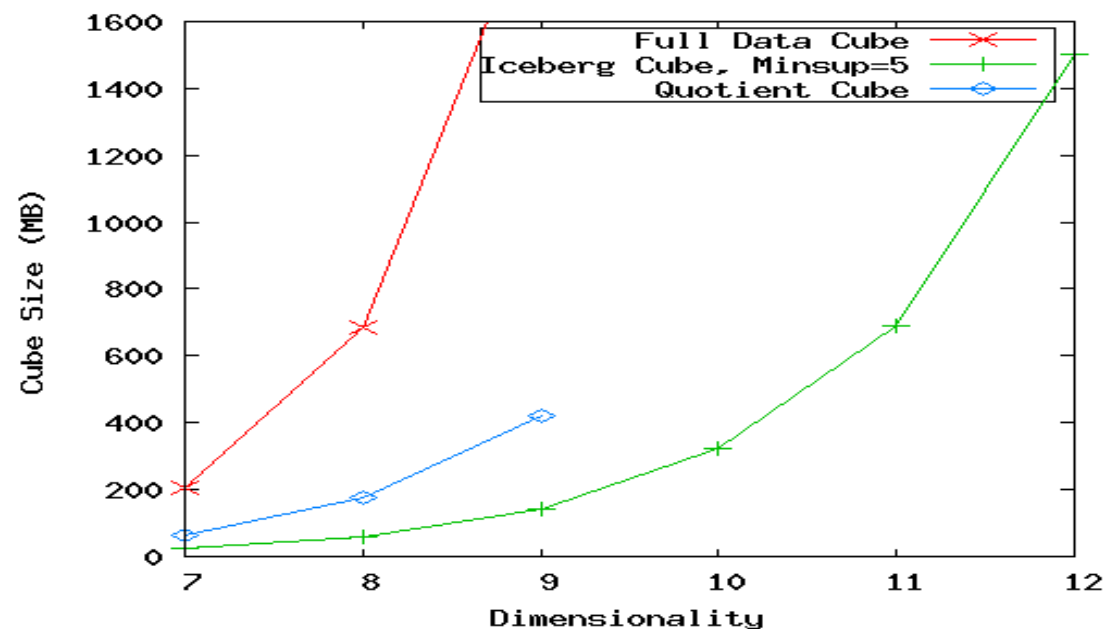
BUC: 分区和聚集

- 通常情况下，整个数据集无法装入主存储器中
- 对不同的值进行排序
 - 分割成适合的区块
- 继续处理
- 优化
 - 分区
 - 排序，哈希，线性排序
 - 首先处理最有区分能力的维
 - 分区越多，剪枝机会越大
 - 维越均匀，剪枝越好



高维的OLAP? --维数灾难

- 高维 OLAP: 在许多应用中都需要
 - 科学和工程分析
 - 生物数据分析: 成千上万的基因
 - 统计调查: 数以百计的变量
- 以前的立方体方法都不能处理高维度的问题!
 - 冰山方块和压缩方块: 只能延迟不可避免的维数爆炸
 - 完全物化: 在磁盘上访问结果的开销仍然很大
- A shell-fragment approach: X. Li, J. Han, and H. Gonzalez, High-Dimensional OLAP: A Minimal Cubing Approach, VLDB'04



带有最小立方体的快速高维OLAP

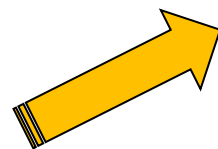
- 观察结果：大部分OLAP操作一次只在少数维上执行
- 半在线计算模型
 - 将维度集划分为外壳片段
 - 计算每个外壳片段的数据立方体，同时保留倒排索引
 - 鉴于预先计算的立方体片段，动态地在线计算高维数据立方体的立方体单元。
- 主要想法：在预计算量和在线计算速度之间进行权衡
 - 将计算高维立方体减少为预先计算一组低维立方体
 - 原始高维空间的在线重构
 - 无损还原

用包含2个维度的外壳片段计算5-D立方体

- 例子：立方体的聚合函数为**count**

<i>tid</i>	A	B	C	D	E
1	a1	b1	c1	d1	e1
2	a1	b2	c1	d2	e1
3	a1	b2	c1	d1	e2
4	a2	b1	c1	d1	e2
5	a2	b1	c1	d1	e3

- 将5-D表格分成2个外壳片段。
 - (A, B, C) 和 (D, E)
- 建立传统的倒排索引或RID列表



属性值	TID列表	列表尺寸
a1	1 2 3	3
a2	4 5	2
b1	1 4 5	3
b2	2 3	2
c1	1 2 3 4 5	5
d1	1 3 4 5	4
d2	2	1
e1	1 2	2
e2	3 4	2
e3	5	1

Shell Fragment Cubes: Ideas

- 在数据立方体意义上，将一维倒置指数泛化为多维指数
- 计算数据立方体ABC和DE的所有立方体，同时保留倒置的指数。
 - 例如：贝壳碎片的立方体ABC包含7个立方体。
 - a, b, c; ab, ac, bc; abc

- 这样就完成了离线计算

ID_Measure表

- 如果使用的度量不是count，则存储在ID_measure表中

tid	count	sum
1	5	70
2	3	10
3	8	20
4	5	40
5	2	30

Shell-fragment AB



单元	交	TID列表	列表长度
a1 b1	$1\ 2\ 3 \cap 1\ 4\ 5$	1	1
a1 b2	$1\ 2\ 3 \cap 2\ 3$	2 3	2
a2 b1	$4\ 5 \cap 1\ 4\ 5$	4 5	2
a2 b2	$4\ 5 \cap 2\ 3$	\varnothing	0

属性值	TID列表	列表尺寸
a1	1 2 3	3
a2	4 5	2
b1	1 4 5	3
b2	2 3	2
c1	1 2 3 4 5	5
d1	1 3 4 5	4
d2	2	1
e1	1 2	2
e2	3 4	2
e3	5	1

Shell Fragment Cubes: Size and Design

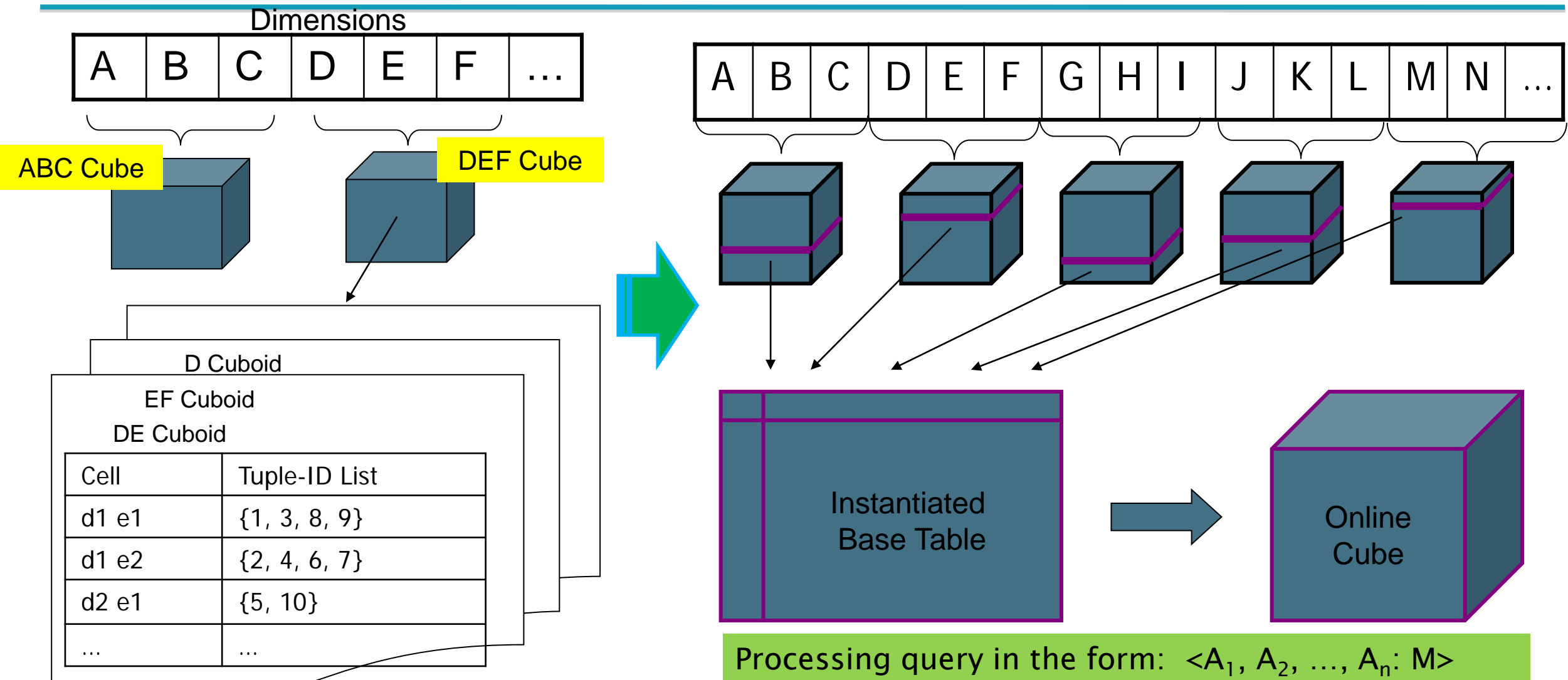
- 给定一个有T个图元、D个维度、F外壳片段大小的数据库，片段立方体的空间需求为：

- 对于 $F < 5$ ，增长是亚线性的 $O\left(T \left\lceil \frac{D}{F} \right\rceil (2^F - 1)\right)$
- 壳的片段不一定是相交的
- 片段分组可以是任意的，以实现最大的在线性能
 - 已知的常见组合（例如，<城市，州>）应被归为一组。
- 壳碎片大小可以调整，以实现离线和在线计算的最佳平衡

属性值	TID列表	列表尺寸
a1	1 2 3	3
a2	4 5	2
b1	1 4 5	3
b2	2 3	2
c1	1 2 3 4 5	5
d1	1 3 4 5	4
d2	2	1
e1	1 2	2
e2	3 4	2
e3	5	1

单元	交	TID列表	列表长度
a1 b1	$1\ 2\ 3 \cap 1\ 4\ 5$	1	1
a1 b2	$1\ 2\ 3 \cap 2\ 3$	2 3	2
a2 b1	$4\ 5 \cap 1\ 4\ 5$	4 5	2
a2 b2	$4\ 5 \cap 2\ 3$	\varnothing	0

使用Frag-Shells进行在线OLAP查询计算

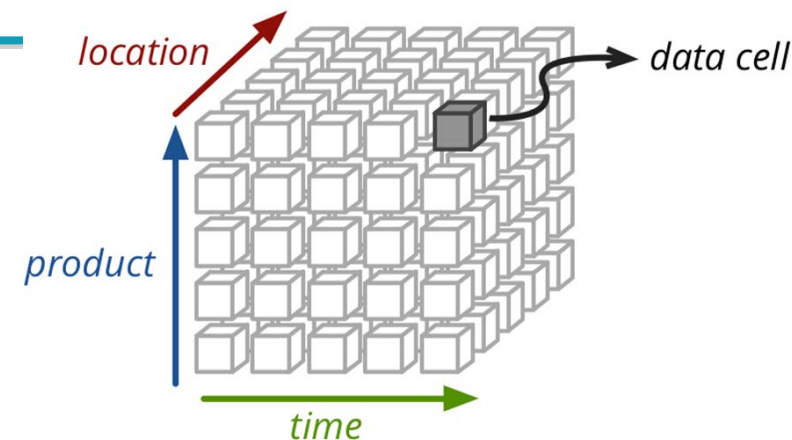


用壳片段进行在线查询计算

- 一个查询的一般形式是： $\langle A_1, A_2, \dots, A_n: M \rangle$ 。
- 每个 A_i 有3个可能的值（例如， $\langle 3, ?, *, 1: \text{count} \rangle$ 返回一个二维数据立方体）
 - 实值
 - 汇总*
 - 询问？
- 方法:给出物化的片段立方体，处理一个查询，如下所示
 - 将查询分成几个片段，与shell-fragment相同
 - 从片段立方体中为每个片段获取相应的TID列表
 - 与每个片段的TID列表相交，构建实例化的基础表
 - 使用基表与任何立方体算法计算数据立方体

第5章：数据立方体技术

- 数据立方体的计算：基本概念
- 数据立方体的计算方法
- 数据立方体空间的多维数据分析
- 摘要



立方体空间的数据挖掘

- 数据立方体极大地开阔了分析的范围
- OLAP-风格的分析和数据挖掘的四种互动方式
 - 使用立方体空间来定义数据空间进行挖掘
 - 使用OLAP查询生成特征和目标进行挖掘，例如多特征立方体
 - 将数据挖掘模型作为多步骤挖掘过程中的构件，例如预测立方体
 - 使用数据立方体计算技术来加快重复模型的构建速度
 - 立体空间数据挖掘可能需要为每个候选数据空间建立一个模型
 - 在不同候选人的模型构建过程中共享计算，增加数据挖掘的效率。

多粒度的复杂聚集：多特征立方体

- 多特征立方体（Ross, et al. 1998）：计算复杂的查询，涉及多个粒度的多个依赖性聚集
- 例子：按{item, region, month}的所有子集分组，找出2010年每组的最高价格，并在具有最高价格的所有元组中找出总销售额

```
select item, region, month, max(price), sum(R.sales)
```

```
from purchases
```

```
where year = 2010
```

```
cube by item, region, month: R
```

```
such that R.price = max(price)
```

发现驱动的数据立方体探查

- 发现驱动的巨大立方体空间的探索（Sarawagi, et al., 98）。
 - 有效导航大型OLAP数据立方体
 - 预先计算显示例外情况的措施，指导用户进行数据分析，在所有层面上进行汇总
 - 例外情况：根据统计模型，与预期值有明显不同
 - 背景颜色等视觉线索被用来反映每个单元的异常程度。
- 异常指示
 - SelfExp: 指示相对于同一聚集层的其他单元，该单元的奇异程度
 - InExp: 指示该单元之下某处的奇异程度，如果从它下钻
 - PathExp: 指示由该单元的每条下钻路径的奇异程度
- 异常指标的计算可以与立方体的构建相叠加
 - 异常情况可以像预先计算的总量一样被存储、索引和检索

实例：发现驱动的数据立方体

item	all
region	all

Sum of sales	month											
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Total		1%	-1%	0%	1%	3%	-1	-9%	-1%	2%	-4%	3%

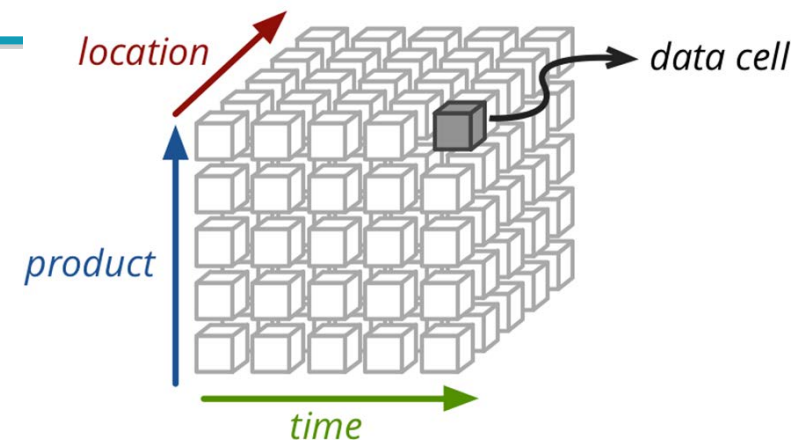
Avg sales	month											
item	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Sony b/w printer		9%	-8%	2%	-5%	14%	-4%	0%	41%	-13%	-15%	-11%
Sony color printer		0%	0%	3%	2%	4%	-10%	-13%	0%	4%	-6%	4%
HP b/w printer		-2%	1%	2%	3%	8%	0%	-12%	-9%	3%	-3%	6%
HP color printer		0%	0%	-2%	1%	0%	-1%	-7%	-2%	1%	-5%	1%
IBM home computer		1%	-2%	-1%	-1%	3%	3%	-10%	4%	1%	-4%	-1%
IBM laptop computer		0%	0%	-1%	3%	4%	2%	-10%	-2%	0%	-9%	3%
Toshiba home computer		-2%	-5%	1%	1%	-1%	1%	5%	-3%	-5%	-1%	-1%
Toshiba laptop computer		1%	0%	3%	0%	-2%	-2%	-5%	3%	2%	-1%	0%
Logitech mouse		3%	-2%	-1%	0%	4%	6%	-11%	2%	1%	-4%	0%
Ergo-way mouse		0%	0%	2%	3%	1%	-2%	-2%	-5%	0%	-5%	8%

item	IBM home computer
------	-------------------

Avg sales	month											
region	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
North		-1%	-3%	-1%	0%	3%	4%	-7%	1%	0%	-3%	-3%
South		-1%	1%	-9%	6%	-1%	-39%	9%	-34%	4%	1%	7%
East		-1%	-2%	2%	-3%	1%	18%	-2%	11%	-3%	-2%	-1%
West		4%	0%	-1%	-3%	5%	1%	-18%	8%	5%	-8%	1%

第5章：数据立方体技术

- 数据立方体的计算：基本概念
- 数据立方体的计算方法
- 数据立方体空间的多维数据分析
- 摘要



数据方块技术：摘要

- 数据立方体的计算：初步的概念
- 数据立方体的计算方法
 - 多路阵列聚合
 - BUC
 - 使用Shell-Fragments的高维OLAP
- 立方空间的多维数据分析
 - 多功能立方体
 - 发现驱动的数据立方体探索

Data Cube Technology: References (I)

- S. Agarwal, R. Agrawal, P. M. Deshpande, A. Gupta, J. F. Naughton, R. Ramakrishnan, and S. Sarawagi. On the computation of multidimensional aggregates. VLDB'96
- K. Beyer and R. Ramakrishnan. Bottom-Up Computation of Sparse and Iceberg CUBEs.. SIGMOD'99
- J. Han, J. Pei, G. Dong, K. Wang. Efficient Computation of Iceberg Cubes With Complex Measures. SIGMOD'01
- L. V. S. Lakshmanan, J. Pei, and J. Han, Quotient Cube: How to Summarize the Semantics of a Data Cube, VLDB'02
- X. Li, J. Han, and H. Gonzalez, High-Dimensional OLAP: A Minimal Cubing Approach, VLDB'04
- X. Li, J. Han, Z. Yin, J.-G. Lee, Y. Sun, "Sampling Cube: A Framework for Statistical OLAP over Sampling Data", SIGMOD'08
- K. Ross and D. Srivastava. Fast computation of sparse datacubes. VLDB'97
- D. Xin, J. Han, X. Li, B. W. Wah, Star-Cubing: Computing Iceberg Cubes by Top-Down and Bottom-Up Integration, VLDB'03
- Y. Zhao, P. M. Deshpande, and J. F. Naughton. An array-based algorithm for simultaneous multidimensional aggregates. SIGMOD'97
- D. Burdick, P. Deshpande, T. S. Jayram, R. Ramakrishnan, and S. Vaithyanathan. OLAP over uncertain and imprecise data. VLDB'05

Data Cube Technology: References (II)

- R. Agrawal, A. Gupta, and S. Sarawagi. Modeling multidimensional databases. ICDE'97
- B.-C. Chen, L. Chen, Y. Lin, and R. Ramakrishnan. Prediction cubes. VLDB'05
- B.-C. Chen, R. Ramakrishnan, J.W. Shavlik, and P. Tamma. Bellwether analysis: Predicting global aggregates from local regions. VLDB'06
- Y. Chen, G. Dong, J. Han, B. W. Wah, and J. Wang, Multi-Dimensional Regression Analysis of Time-Series Data Streams, VLDB'02
- R. Fagin, R. V. Guha, R. Kumar, J. Novak, D. Sivakumar, and A. Tomkins. Multi-structural databases. PODS'05
- J. Han. Towards on-line analytical mining in large databases. SIGMOD Record, 27:97–107, 1998
- T. Imielinski, L. Khachiyan, and A. Abdulghani. Cubegrades: Generalizing association rules. Data Mining & Knowledge Discovery, 6:219–258, 2002.
- R. Ramakrishnan and B.-C. Chen. Exploratory mining in cube space. Data Mining and Knowledge Discovery, 15:29–54, 2007.
- K. A. Ross, D. Srivastava, and D. Chatziantoniou. Complex aggregation at multiple granularities. EDBT'98
- S. Sarawagi, R. Agrawal, and N. Megiddo. Discovery-driven exploration of OLAP data cubes. EDBT'98
- G. Sathe and S. Sarawagi. Intelligent Rollups in Multidimensional OLAP Data. VLDB'01