数据库系统之四
--数据库管理系统实现技术

第20讲 数据库查询实现算法-II

(两趟扫描算法)

战德臣

哈尔滨工业大学　教授.博士生导师

黑龙江省教学名师

教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent Computing for Enterprises & Services, Harbin Institute of Technology

本讲学习什么?

战德臣 教授

基本内容

1. 为什么需要两趟算法暨两趟算法的基本思想
2. 两阶段多路归并排序算法
3. 基于排序的两趟扫描算法
4. 基于散列的两趟扫描算法

重点与难点

- 理解两趟算法的基本思想
- 理解两阶段多路归并排序算法，进一步理解基于排序的两趟扫描算法
- 理解散列算法的核心思想，进一步理解基于散列的两趟扫描算法
- 关系代数操作的两趟扫描算法实现

为什么需要两趟算法?

战德臣

哈尔滨工业大学　教授.博士生导师

黑龙江省教学名师

教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent Computing for Enterprises & Services, Harbin Institute of Technology

为什么需要两趟算法?
(1)整个关系操作存在的问题?

战德臣 教授

数据库的三大类操作

- 一次单一元组的一元操作
  - ✓ $\sigma_F(R)$, $\pi_\alpha(R)$ ---SELECTION, PROJECTION
- 整个关系的一元操作
  - ✓ $\delta(R)$, $\gamma(R)$, $\tau(R)$ ---DISTINCT, GROUP BY，SORTING
- 整个关系的二元操作
  - ✓ 集合上的操作：$\cup_S$ ，$\cap_S$，$-_S$
  - ✓ 包上的操作：$\cup_B$ ，$\cap_B$，$-_B$
  - ✓ 积，连接：PRODUCT， JOIN

迭代器算法

一趟扫描算法　两趟扫描算法　多趟扫描算法

基于排序的算法　基于散列的算法　基于索引的算法

战德臣 教授

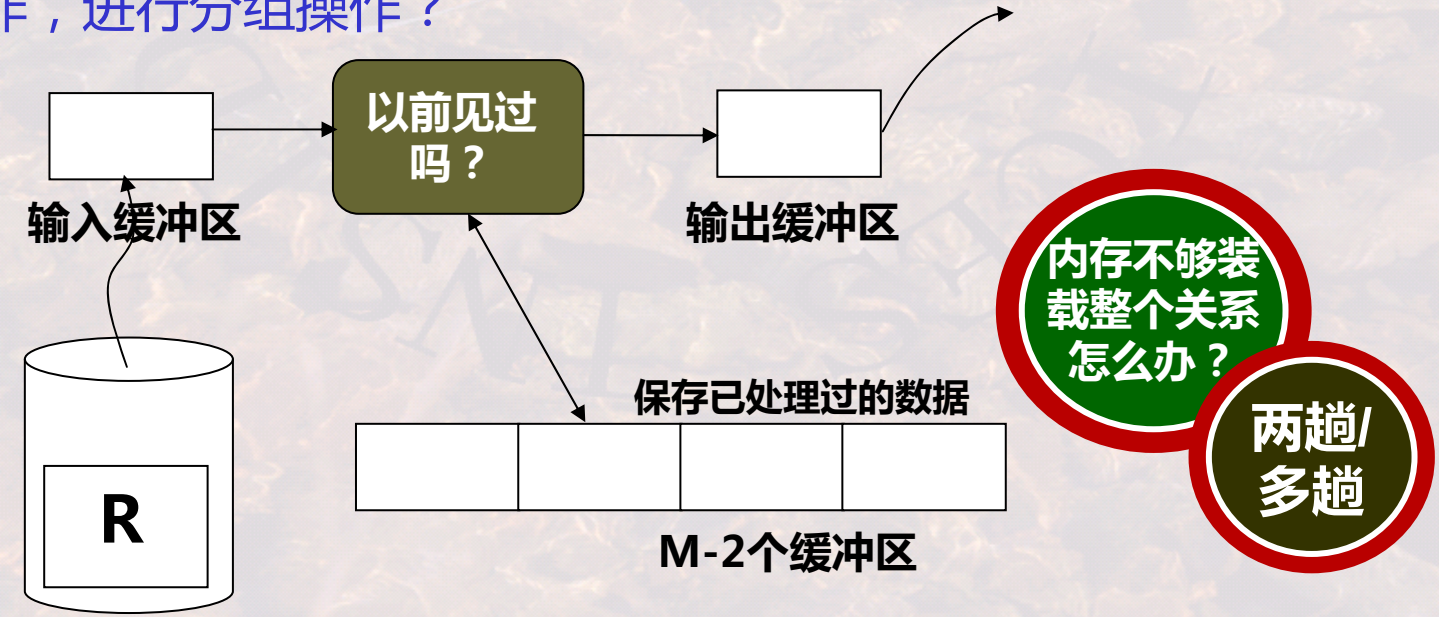
整个关系的一元操作

✓δ(R), γ(R), τ(R) ---DISTINCT, GROUP BY，SORTING

●理论上，任何一个元组需要**与所有元组**进行比较，才能确定是否重复，才能知道是否是一个新的组，才能确定位于何序位置? 这些需要内存

●如果需保存的待处理数据块数远远大于内存可用块数时？怎么办？

●例如：内存只有8块，如何排序70块的数据集？如何针对70块的数据集进行去重复操作，进行分组操作？

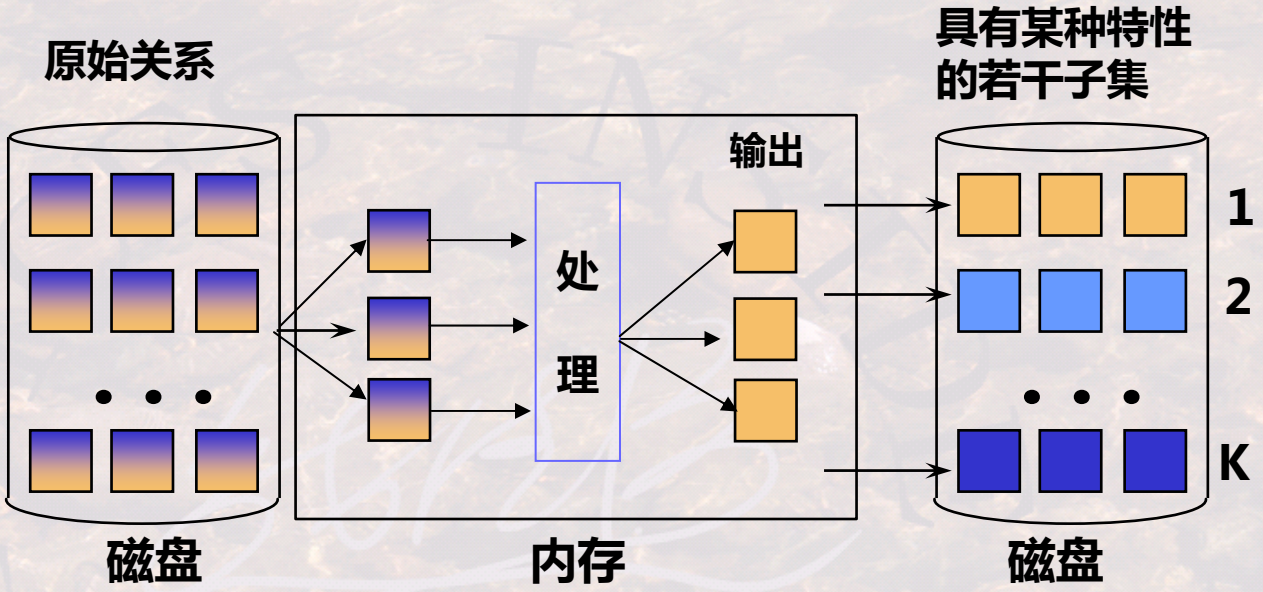


为什么需要两趟算法?
(2)两趟算法的基本思路?

战德臣 教授

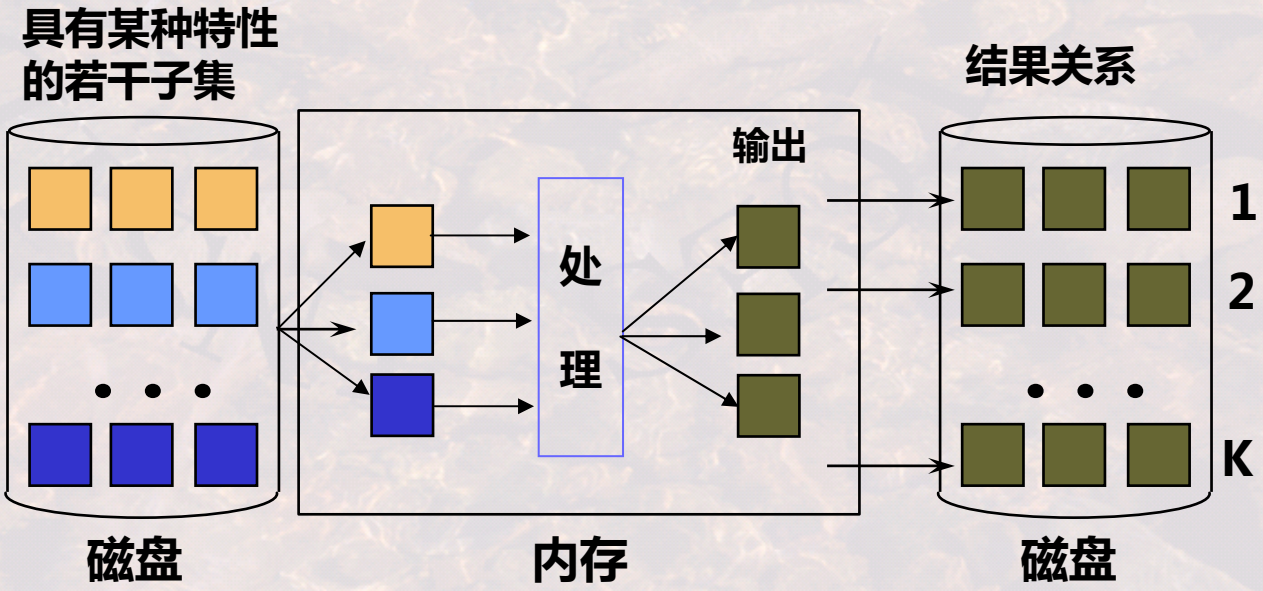
第一趟

划分子集，并使子集具有某种特性，如有序或相同散列值等



处理全局性内容的操作，形成结果关系。如多子集间的归并排序，相同散列值子集的操作等

第二趟



为什么需要两趟算法?
(2)两趟算法的基本思路?

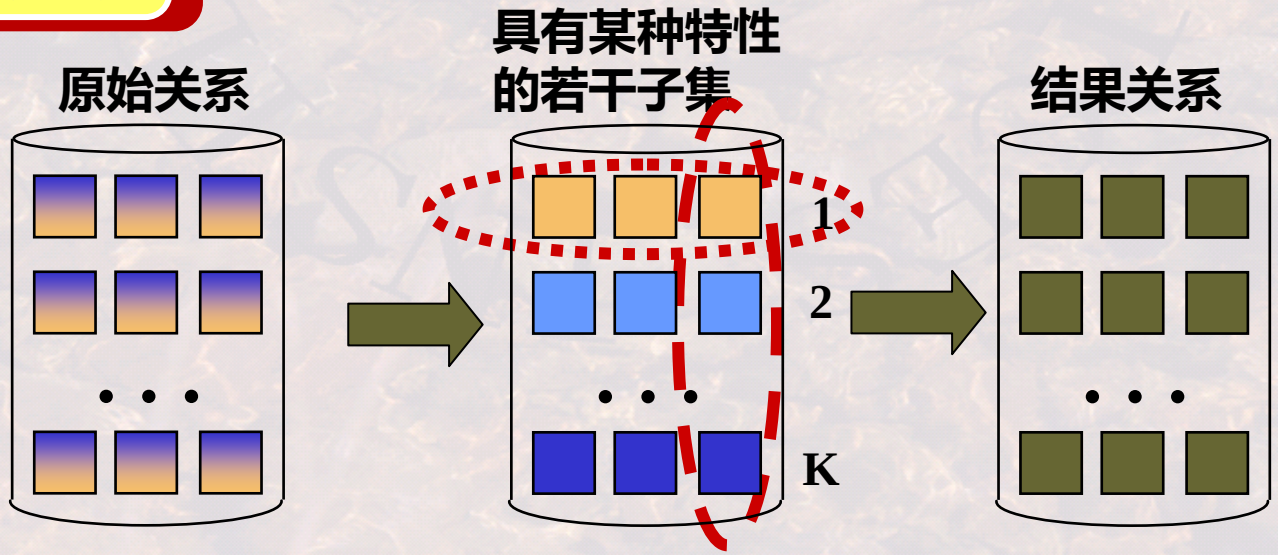
战德臣 教授

基于散列的两趟算法

大数据集上的操作 可否 等于 (子集上操作)的并集?
例如:元组在某一子集上无重复即相当于在全集上无重复

多个已按横向处理的子集上，纵向归并结果等同于在全集上的处理结果?
例如:多个已排序的子集，纵向归并的结果等同于全集上的排序结果
多个已排序的子集，纵向归并处理的无重复即等同于全集上的无重复

基于排序的两趟算法



两阶段多路归并排序
(Two-Phase, Multiway Merge-Sort, TPMMS)

战德臣

哈尔滨工业大学　教授.博士生导师
黑龙江省教学名师
教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent
Computing for Enterprises & Services,
Harbin Institute of Technology

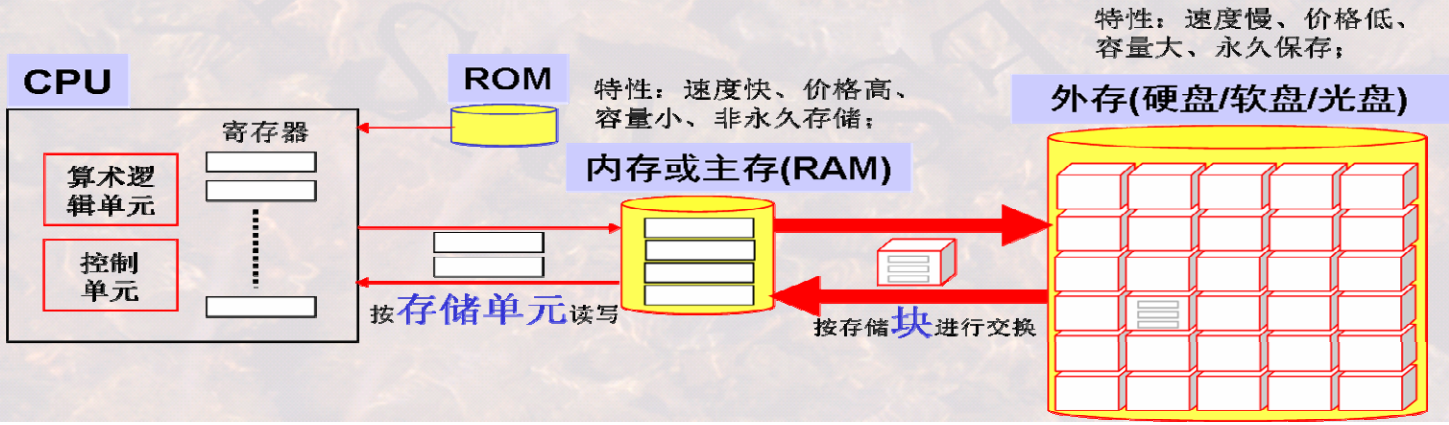
两阶段多路归并排序TPMMS
(1)内排序和外排序问题?

战德臣 教授

●**内排序问题:** 待排序的数据可一次性地装入内存中，即排序者可以完整地看到和操纵所有数据。内存中数据的排序算法：**插入排序算法、选择排序算法、冒泡排序算法**，...(可参阅《数据结构》《高级语言与程序设计》《算法设计与分析》等课程学习相关内容)。

●**外排序问题:** 待排序的数据不能一次性装入内存，即排序者不能一次完整地看到和操纵所有数据，需要将数据分批装入内存分批处理的排序问题；

示例：内存--2GB；待排序数据--7GB, 10GB, 100GB? 更大数据集合这种需要使用硬盘等外部存储设备进行大数据集合排序的过程/算法称为外排序(External sorting)。



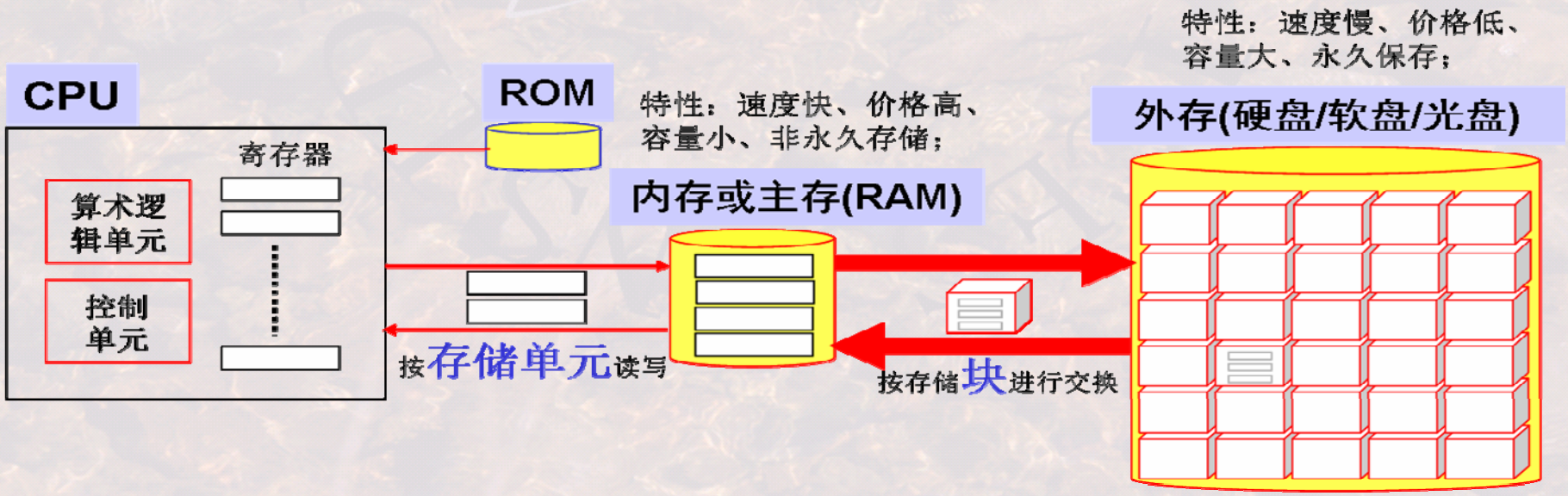
两阶段多路归并排序TPMMS
(2)外排序问题分析?

战德臣 教授

外排序问题分析(仅介绍思想,忽略一些细节), 假设:

- 读写磁盘块函数: ReadBlock, WriteBlock
- 内存大小: 共$B_{memory}=6$块, 每块可装载$R_{block}=5$个元素
- 待排序数据: $R_{problem}=60$个元素, 共占用$B_{problem}=12$块

问题: $B_{problem}$块的数据怎样利用B_{memory}块的内存进行排序?



两阶段多路归并排序TPMMS
(2)外排序问题分析?

战德臣 教授

基本排序策略

●$B_{problem}$块数据可划分为N个子集合，使每个子集合的块数小于内存可用块数，即：$B_{problem}/N < B_{memory}$。每个子集合都可装入内存并采用内排序算法排好序并重新写回磁盘。

问题转化为: N个已排序子集合的数据怎样利用内存进行总排序?

两阶段多路归并排序算法TPMMS

子集合1	90	86	82	80	75	70	60	58	43	32	28	15	11	08	05
子集合2	55	45	35	30	27	24	20	18	10	09	08	08	06	04	03
子集合3	80	78	72	62	52	50	42	38	34	31	29	19	16	13	10
子集合4	72	70	68	64	62	60	45	42	38	35	25	20	15	09	08

B_{memory} =6块,

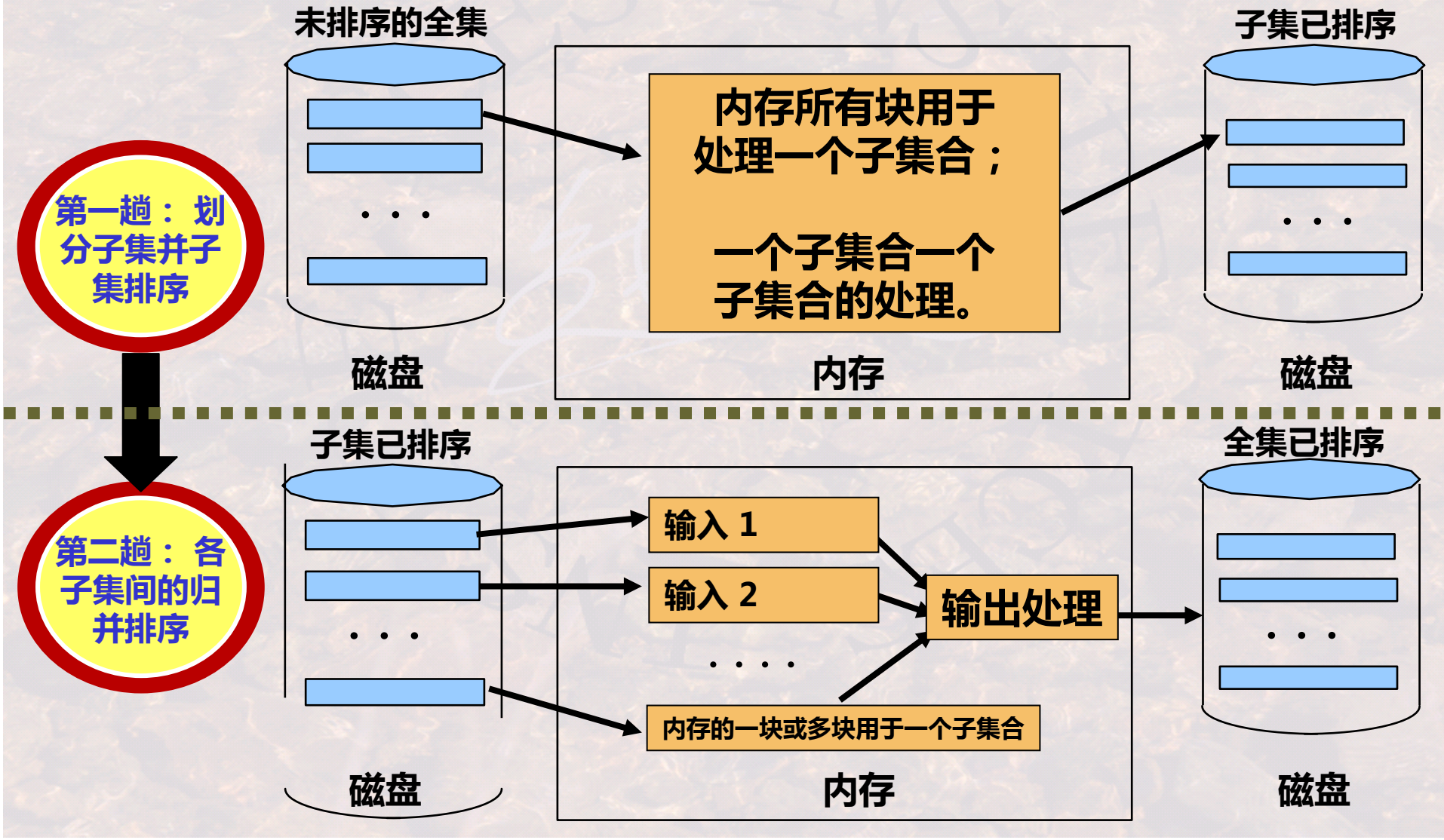
R_{block} =5个元素

$R_{problem}$=60个元素

$B_{problem}$=12块

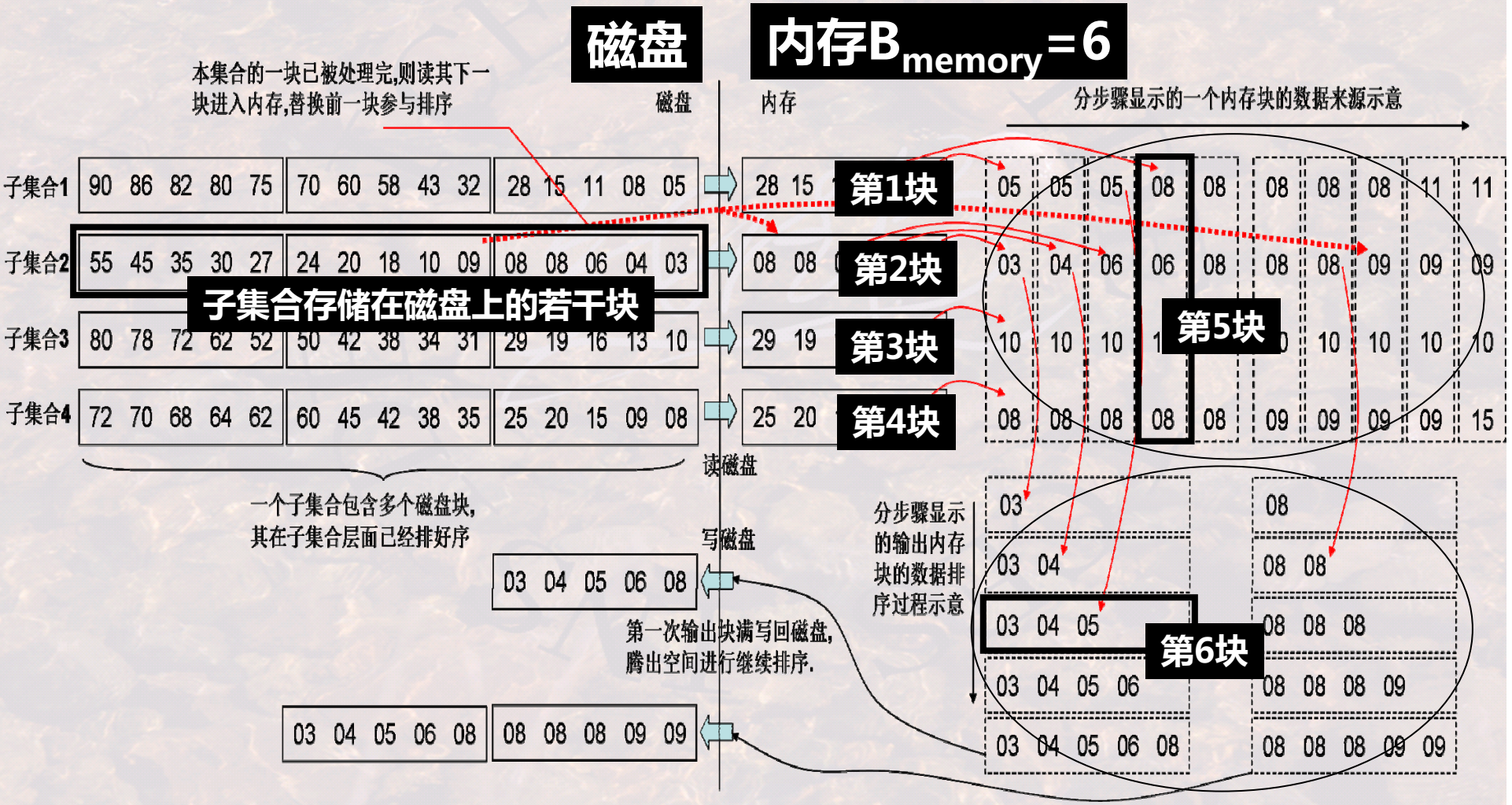
战德臣 教授

两阶段多路归并排序TPMMS(Two-Phase Multiway Merge-Sort)



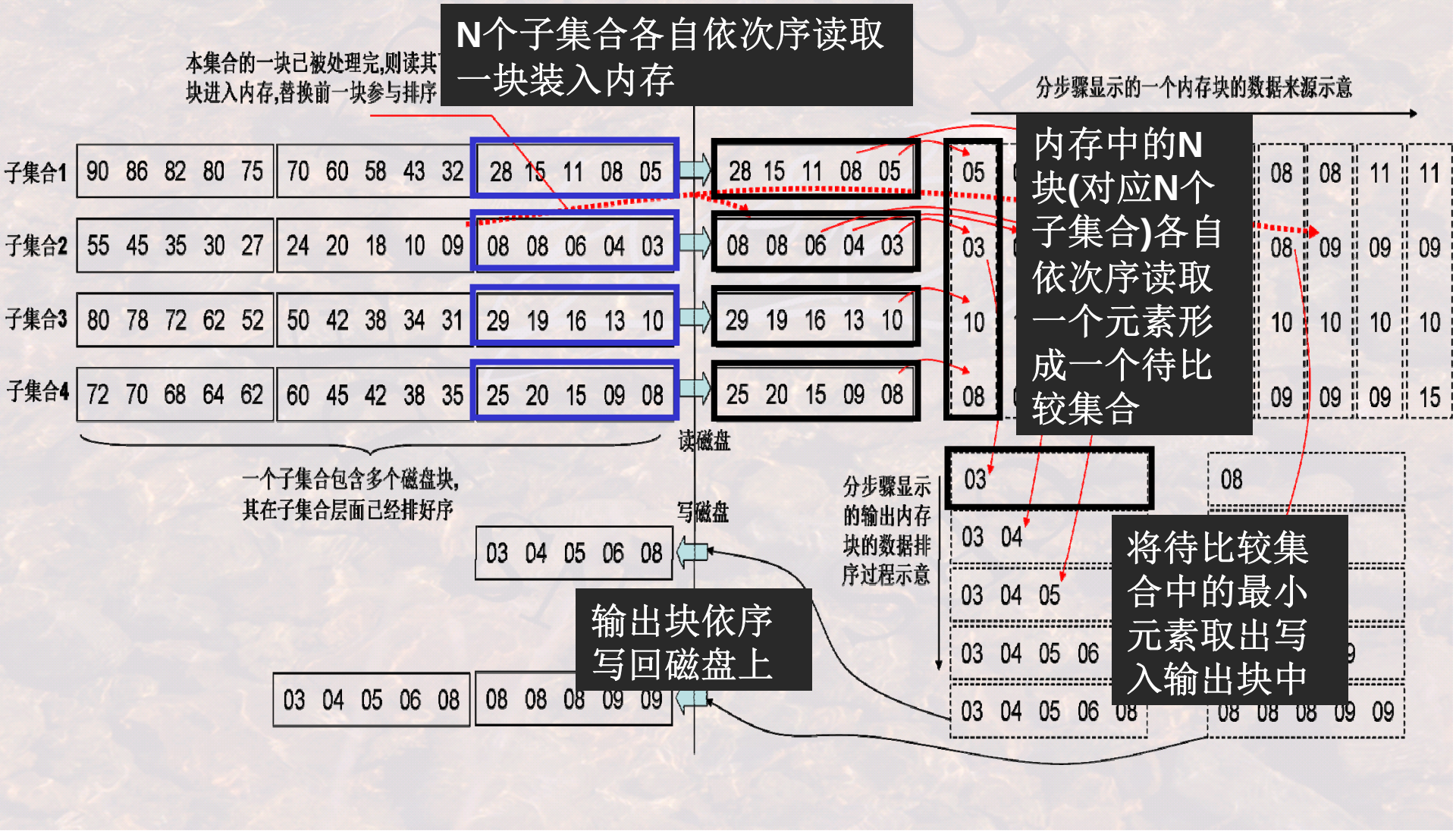
战德臣 教授

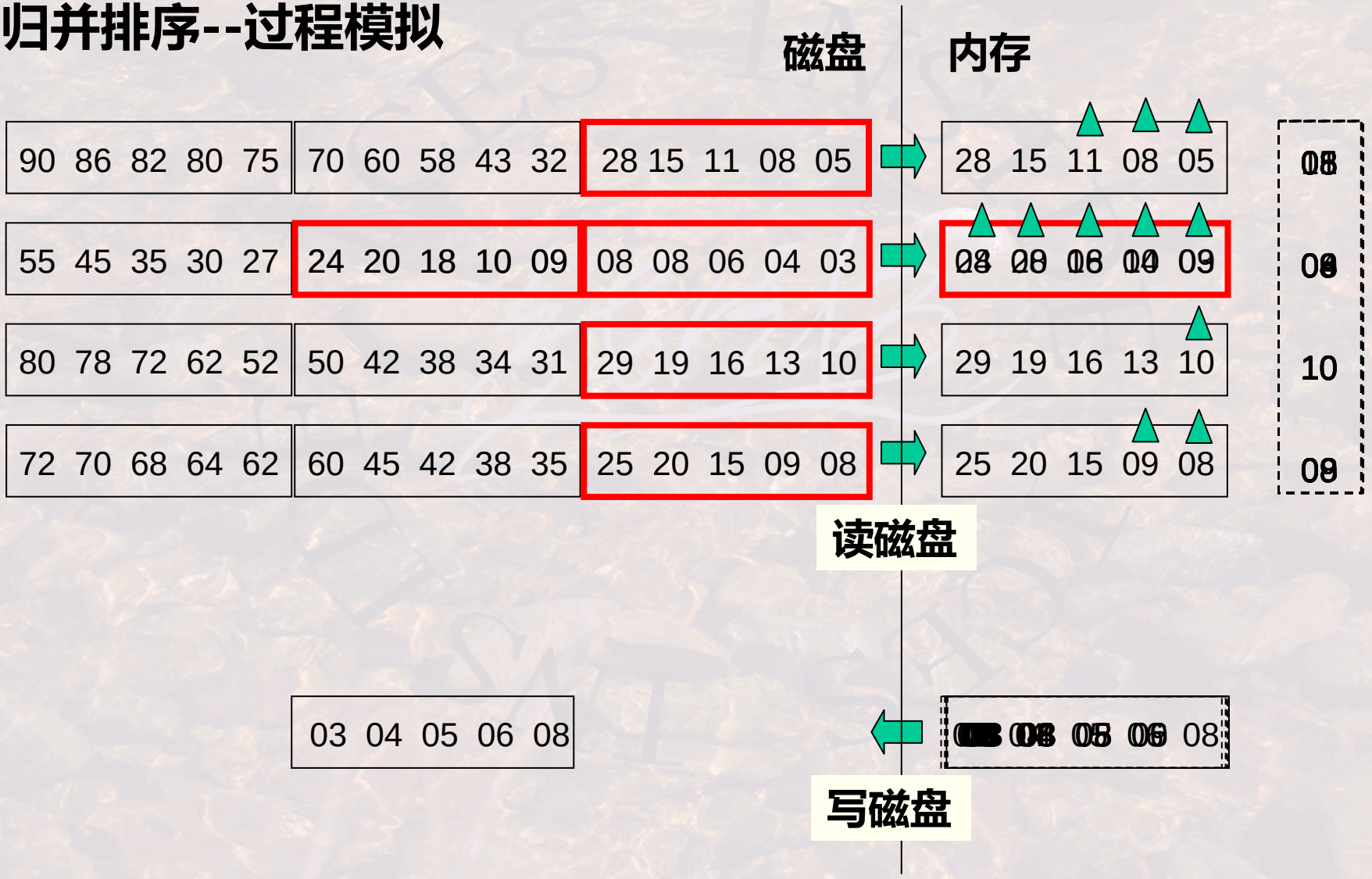
算法对内存资源的使用



两阶段多路归并排序TPMMS
(4)算法求解示例?

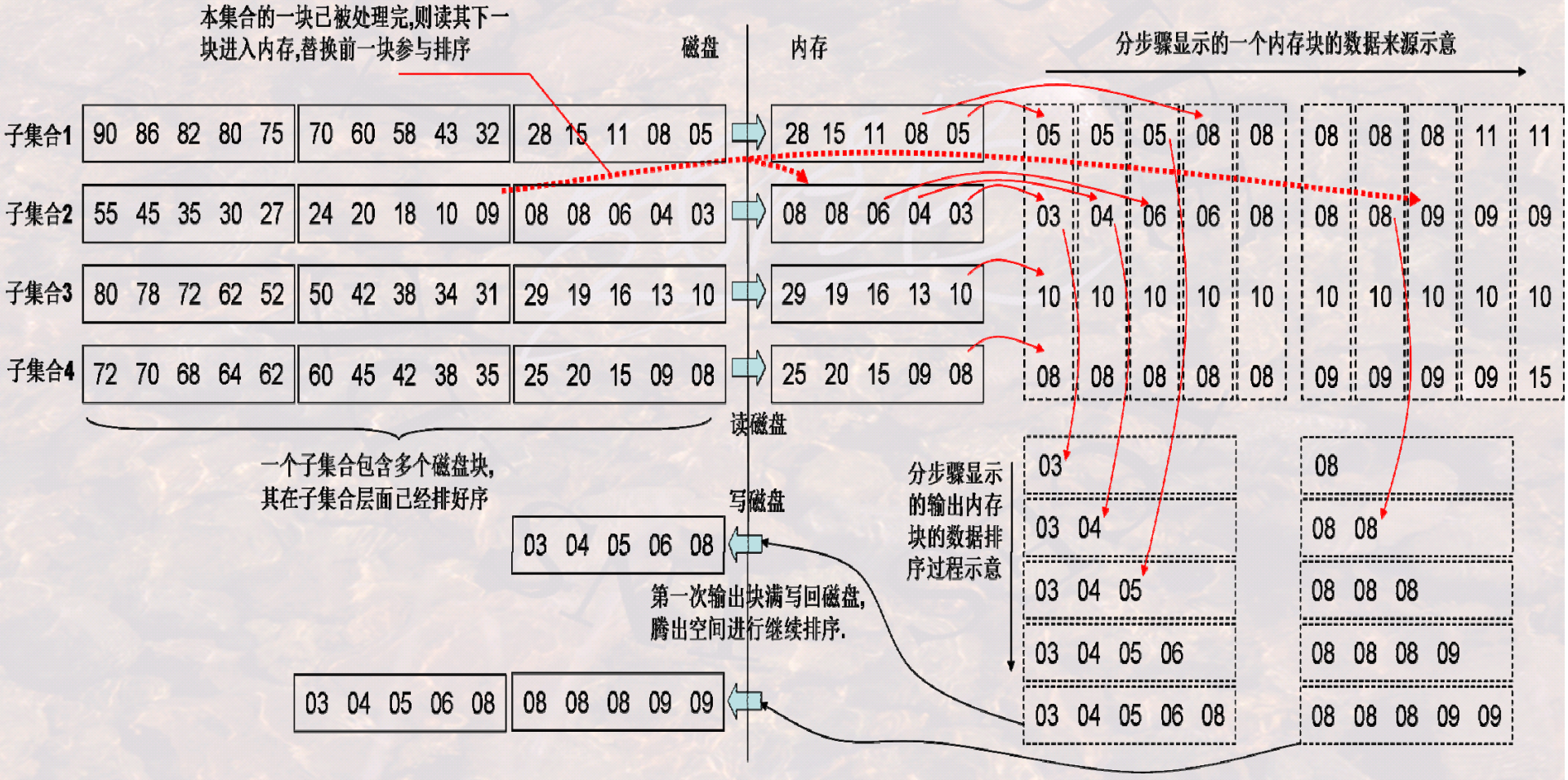
战德臣 教授




归并排序--过程模拟
磁盘
内存
90 86 82 80 75
70 60 58 43 32
28 15 11 08 05
28 15 11 08 05
55 45 35 30 27
24 20 18 10 09
08 08 06 04 03
80 78 72 62 52
50 42 38 34 31
29 19 16 13 10
29 19 16 13 10
72 70 68 64 62
60 45 42 38 35
25 20 15 09 08
25 20 15 09 08
读磁盘
03 04 05 06 08
写磁盘

战德臣 教授

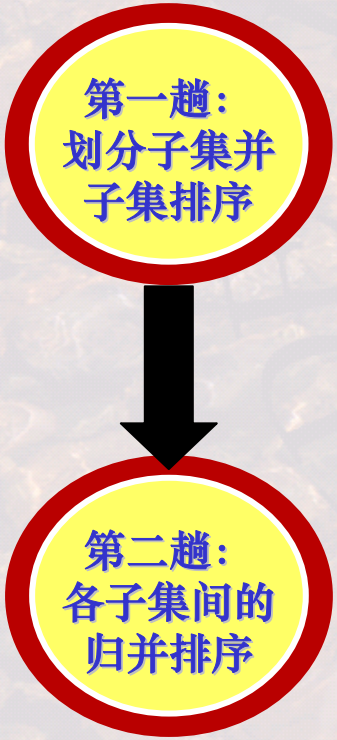
归并排序--过程模拟



两阶段多路归并排序TPMMS
(5)算法表达

战德臣 教授

归并排序--算法描述
(同学自己阅读)



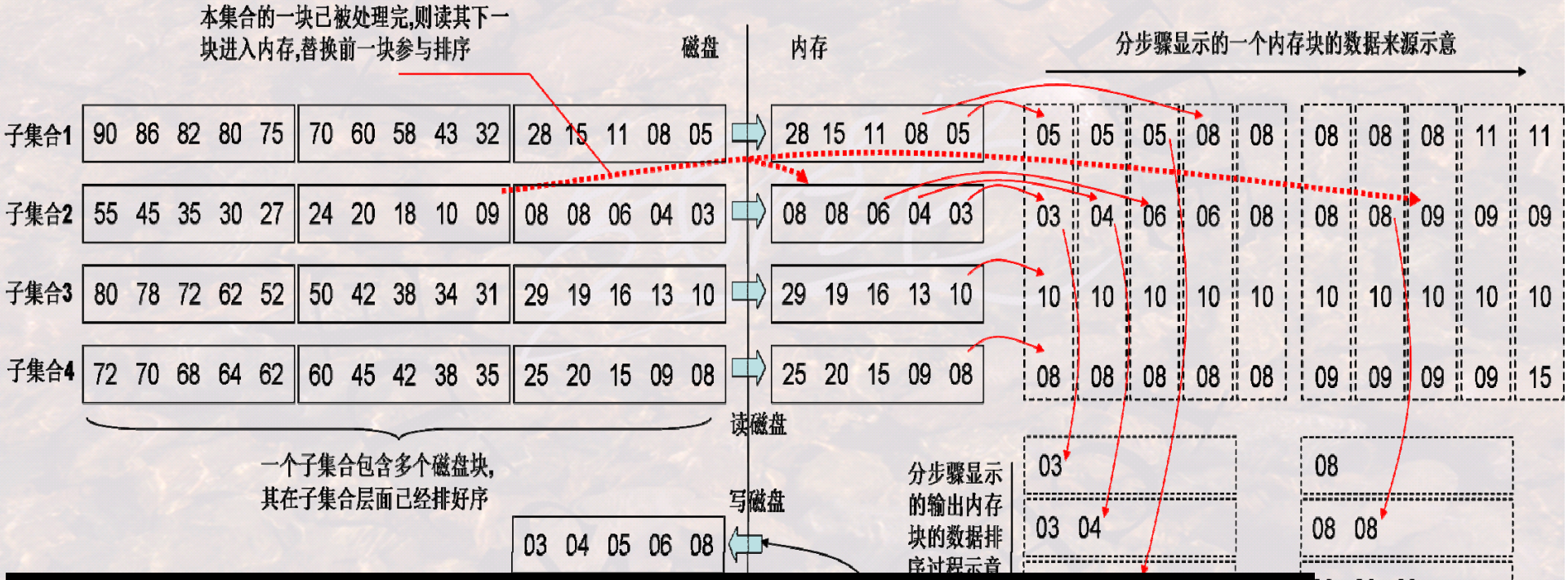
已知：$S_{problem}$为待排序元素集合，$R_{problem}$–待排序集合中的元素个数，R_{block}-磁盘块或内存块能存储的元素个数，B_{memory}-可用内存块的个数，R(S)为求集合S的元素个数的函数，M_i为内存的第i块，P_{output}为输出块内存中当前元素的指针。

1. 将待排序集合 $S_{problem}$划分为 m 个子集合 S_1, S_2,...,S_m, 其中 $S_{problem}=\cup_{i=1,...,m}S_i$，且 $R_{problem}=\Sigma_{i=1,...,m}R(S_i)$, $R(S_i)<=B_{memory}*R_{block}$, i=1,...,m(注:每个 S_i的元素个数小于内存所能装载的元素个数).
2. **for** i=1 **to** m
3. {将 S_i装入内存，并采用一种内排序算法进行排序，排序后再存回相应的外存中}
   /*步骤 2 和 3 完成子集合的排序。接下来要进行归并，M_1,...,M_m用于分别装载 S_1,...,S_m的一块*/;
4. **for** i=1 **to** m
5. { 调用 read block 函数，读 S_i的第一块存入 M_i中，同时将其第一个元素存入 $M_{compare}$的第 i_{th}个位置; }
6. 设置 P_{output}为输出内存块的起始位置;
7. 求 $M_{compare}$中 m 个元素的最小值及其位置 i。
8. **If** (找到最小值及其位置 i) **then**
9. {　　将第 i_{th}个位置的元素存入 M_{output}中的 P_{output}位置，P_{output}指针按次序指向下一位置；
10. 　　**If** (P_{output}指向结束位置) **then**
11. 　　{　调用 Write Block 按次序将 M_{output}写回磁盘；置 P_{output}为输出内存块的起始位置；继续进行; }
12. 　　获取 M_i的下一个元素.
13. 　　**If** (M_i有下一个元素)
14. 　　{ 将 M_i下一个元素存入 $M_{compare}$的第 i_{th}个位置。转步骤 7 继续执行。 }
15. 　　**Else** { 调用 read block 按次序读 S_i的下一块并存入 M_i;
16. 　　**If** (S_i有下一块)
　　{ 将其第一个元素存入 $M_{compare}$的第 i_{th}个位置。转步骤 7 继续执行。 }
17. 　　**ELSE** { 返回一个特殊值如 Finished，以示 S_i子集合处理完毕，M_i为空，且使 $M_{compare}$中的第 i_{th}位置为该特殊值，表明该元素不参与 $M_{compare}$的比较操作。转步骤 7 继续执行。 }
18. 　　}
19. } /*若 $M_{compare}$的所有元素都是特殊值 Finished，即没有最小值，则算法结束*/

两阶段多路归并排序TPMMS
(6)算法相关的讨论话题?

战德臣 教授

算法的复杂性：3 B(R)--不考虑最终结果的写回；4 B(R)—考虑最终结果的写回



算法的效率：读写磁盘块的次数，即I/O数= 4 $B_{problem}$

■子集合排序阶段读一遍写一遍 2 $B_{problem}$

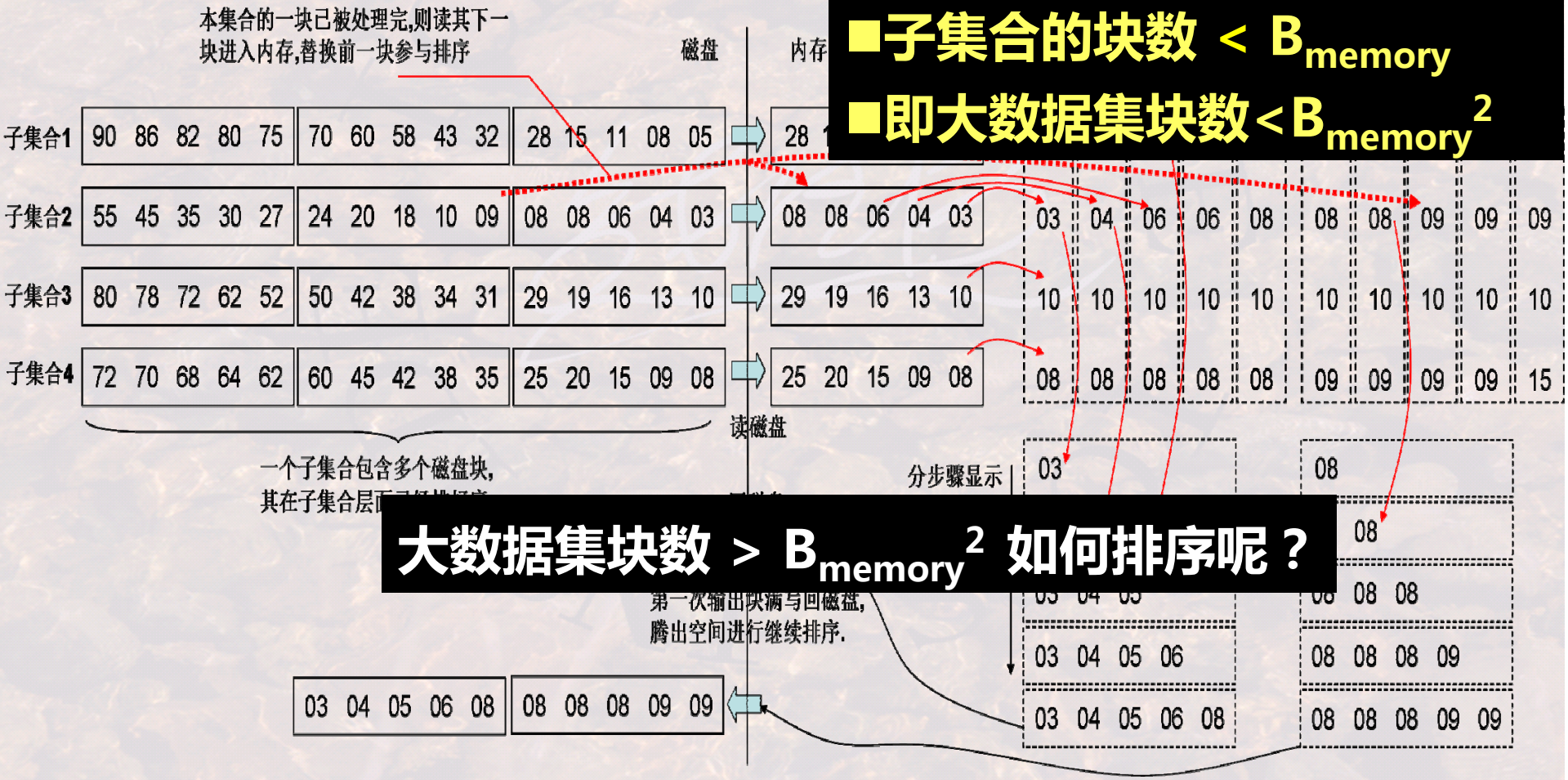
■归并阶段读一遍写一遍 2 $B_{problem}$

战德臣 教授

算法的应用条件

算法的应用条件：

- **子集合数< B_{memory}**
- **子集合的块数 < B_{memory}**
- **即大数据集块数<B_{memory}^2**



大数据集块数 > B_{memory}^2 如何排序呢？

战德臣 教授

更大规模数据集的排序问题—多趟/多阶段

- 内存大小: 共$B_{memory}=3$块
- 待排序数据: 共占用$B_{problem}=30$块

基本策略:

- 30块的数据集 ➔ 10个子集合，每个子集合3块，排序并存储。
- 10个已排序子集合分成5个组：每个组2个子集合, 分别进行二路归并，则可得到5个排好序的集合；
- 5个集合再分成3个组：每个组2个子集，剩余一个单独1组，分别进行二路归并，可得3个排好序的集合；再分组，再归并得到2个排好序的集合；再归并便可完成最终的排序。

思考题

假如内存共有8块，问其如何排序有70块的数据集呢？你是采用二路归并、三路归并、…、七路归并？你设计的具体算法，磁盘读写次数是多少呢？磁盘读写次数最少的应是几路归并？

基于排序的两趟扫描算法

战德臣

哈尔滨工业大学　教授.博士生导师

黑龙江省教学名师

教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent Computing for Enterprises & Services, Harbin Institute of Technology

基于排序的两趟扫描算法
(1)拟解决的问题?

战德臣 教授

数据库的三大类操作

- 一次单一元组的一元操作
  - ✓ $\sigma_F(R)$, $\pi_\alpha(R)$ ---SELECTION, PROJECTION

迭代器算法

- 整个关系的一元操作
  - ✓ $\delta(R)$, $\gamma(R)$, $\tau(R)$ ---DISTINCT, GROUP BY，SORTING
- 整个关系的二元操作
  - ✓ 集合上的操作：$\cup_S$ ，$\cap_S$，$-_S$
  - ✓ 包上的操作：$\cup_B$ ，$\cap_B$，$-_B$
  - ✓ 积，连接：PRODUCT， JOIN

一趟扫描算法　两趟扫描算法　多趟扫描算法

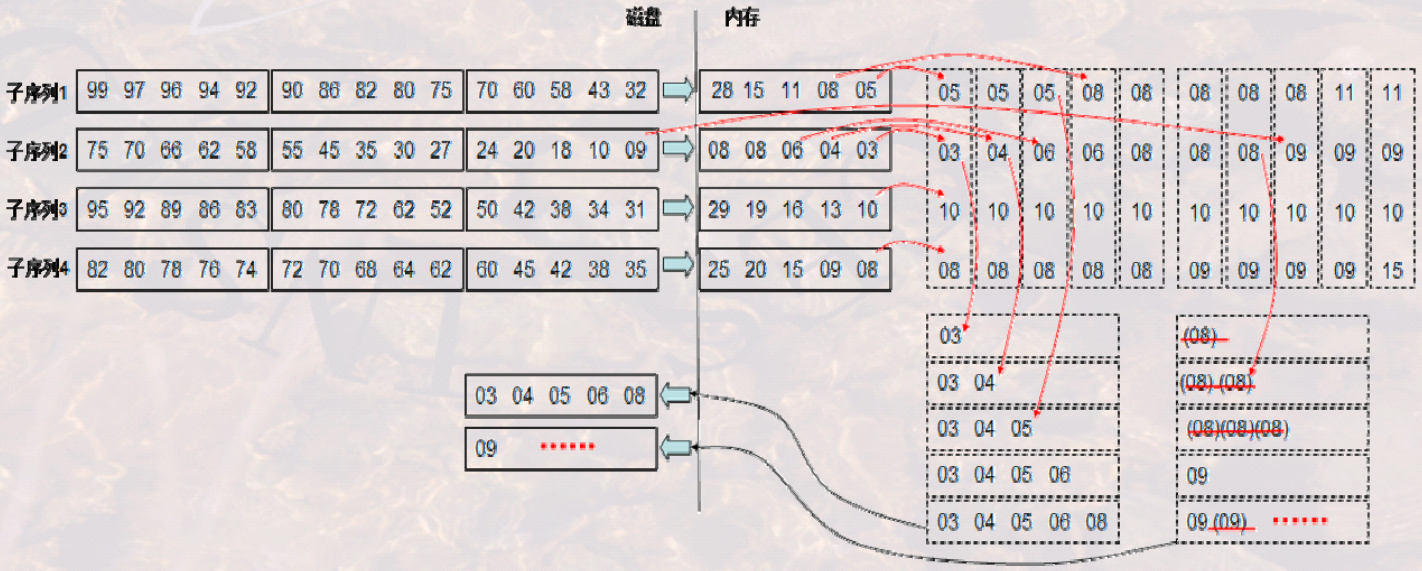
基于排序的算法　基于散列的算法　基于索引的算法

基于排序的两趟扫描算法
(2)去重复操作?

战德臣 教授

$\delta(R)$ --DISTINCT

- 第一趟：划分子表，并进行子表排序
- 第二趟：归并阶段，在排序的基础上，直接将重复的记录剔出掉-不输出
- 算法的复杂性，同TPMMS: 3B(R)---不考虑输出；4B(R)---考虑输出。

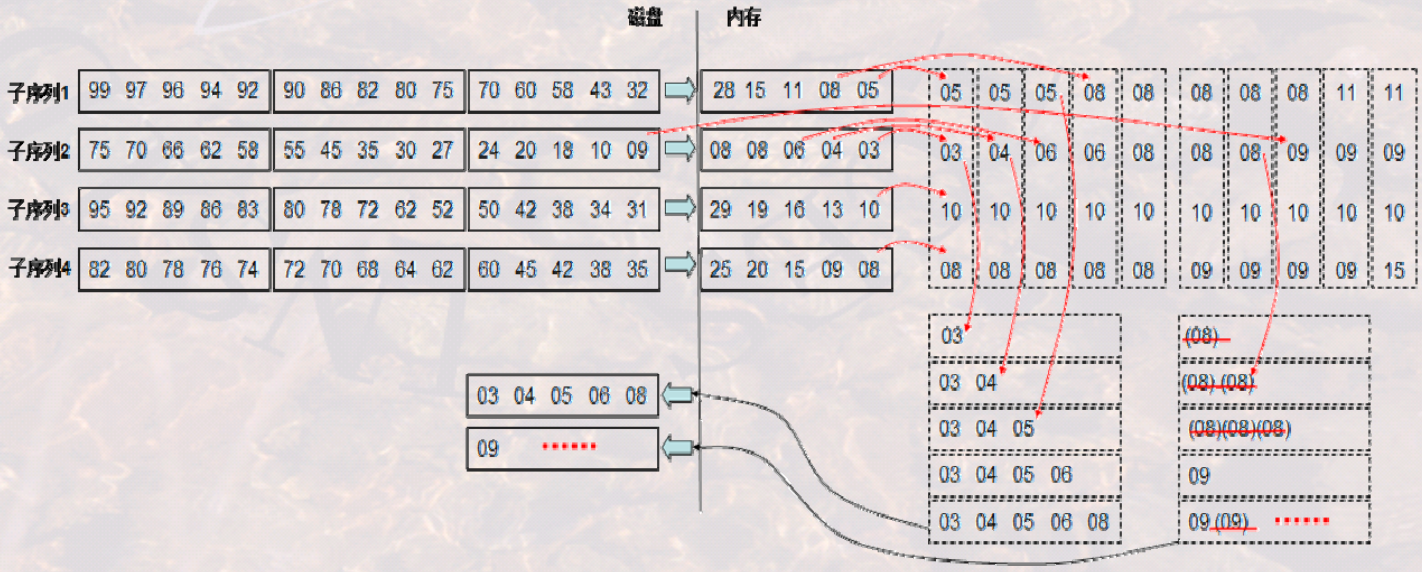


基于排序的两趟扫描算法
(3)分组聚集操作?

战德臣 教授

γ(R) --- GROUP BY

- 第一趟：划分子表，并进行子表排序
- 第二趟：归并阶段，在排序基础上，将不重复的记录，作为新分组输出；将重复的记录进行分组聚集计算。
- 算法的复杂性，同TPMMS: 3B(R)---不考虑输出；4B(R)---考虑输出。

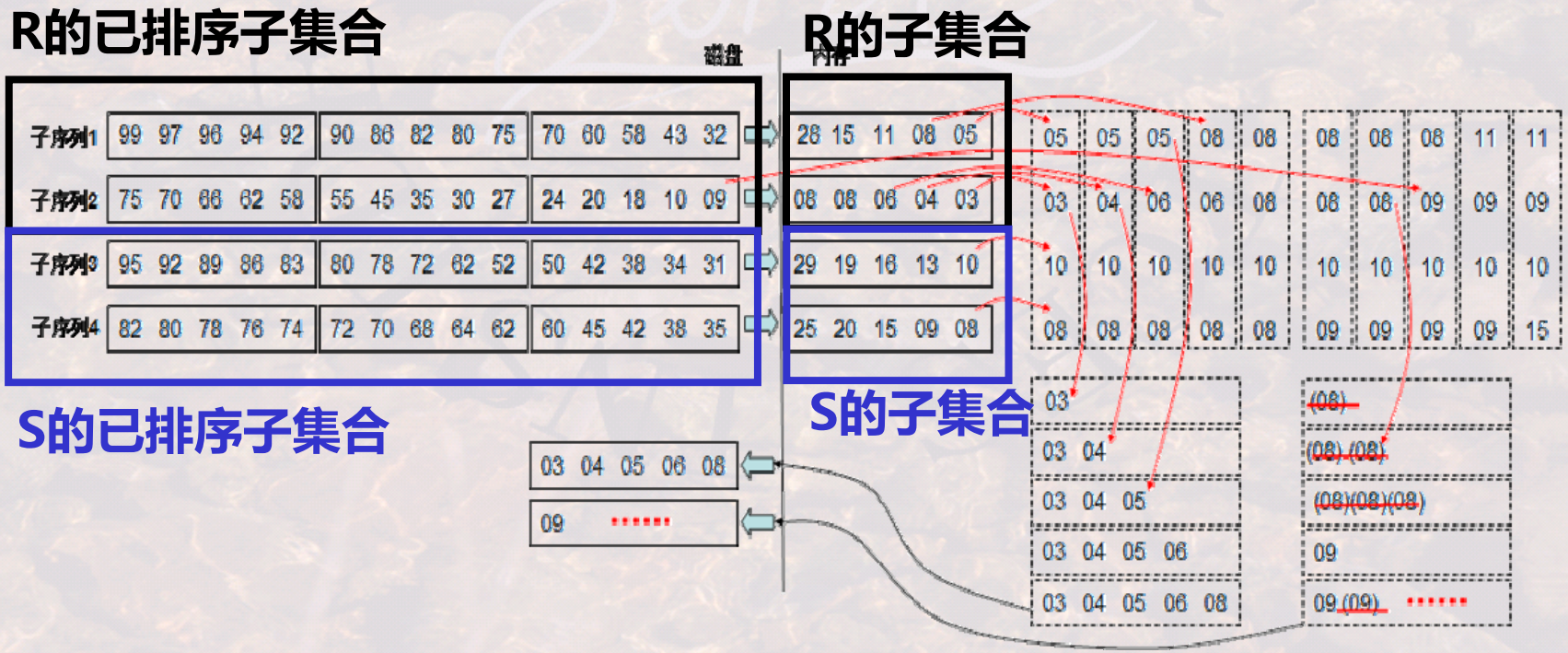


基于排序的两趟扫描算法
(4)基于排序的并、交和差?

战德臣 教授

集合上的操作：$\cup_S$ ，$\cap_S$ ，$-_S$　包上的操作：$\cup_B$ ，$\cap_B$ ，$-_B$

- $\cup_B$ 无需两趟，直接两个关系合并即可。
- $\cup_S$ 需两趟，需要去重复。
- 第一趟：划分R和S的子表并子表排序；第二趟：归并时注意是R的输入还是S的输入。R和S两路输入之间去重复性合并输出。



基于排序的两趟扫描算法
(4)基于排序的并、交和差?

战德臣 教授

集合上的操作：$\cup_S$ ，$\cap_S$ ，$-_S$　包上的操作：$\cup_B$ ，$\cap_B$ ，$-_B$

- $\cap_S$和 $\cap_B$都需要两趟，需要处理出现次数或者去重复。
- 第一趟：划分R和S的子表并子表排序；第二趟：归并时注意是R的输入还是S的输入。R和S的两路输入之间按要求进行输出
  - ✓对于集合交：如果t在R和S中都出现就输出
  - ✓对于包交：输出t的次数是它在R和S中出现的最小次数

基于排序的两趟扫描算法
(4)基于排序的并、交和差?

战德臣 教授

集合上的操作：$\cup_S$ ，$\cap_S$ ，$-_S$　包上的操作：$\cup_B$ ，$\cap_B$ ，$-_B$

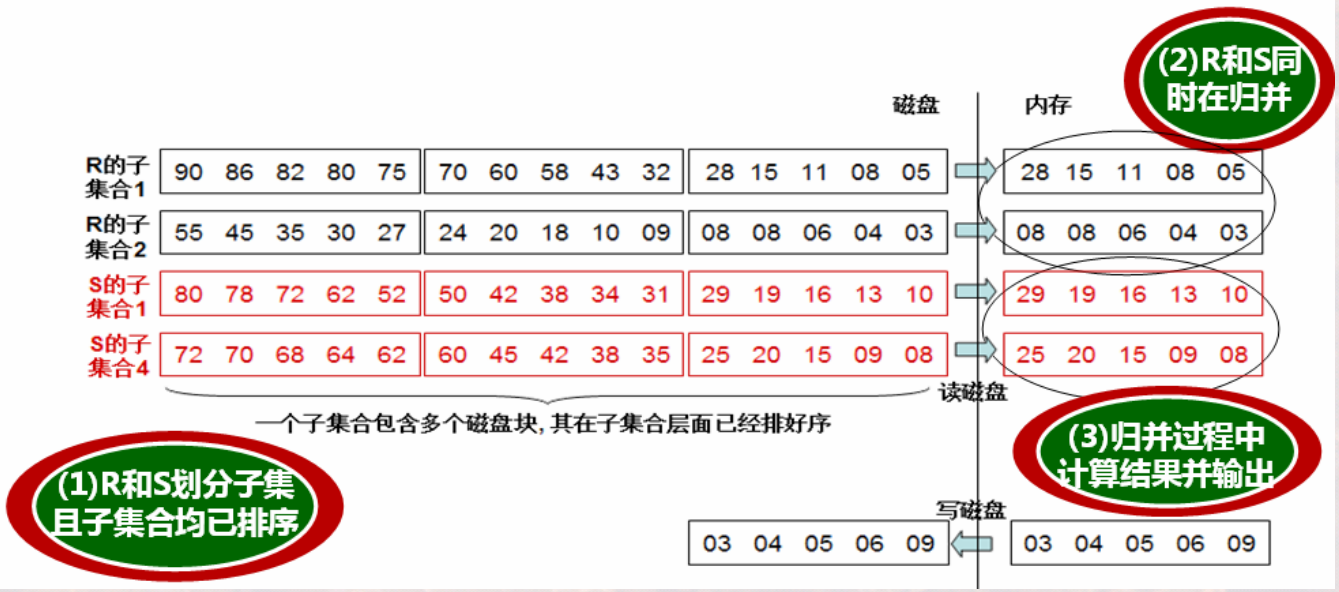
- $-_S$和 $-_B$都需要两趟，需要处理出现次数或者去重复。
- 第一趟：划分R和S的子表并子表排序；第二趟：归并时注意是R的输入还是S的输入。R和S的两路输入之间按要求进行输出
  - ✓对于集合差：R $-_S$ S，当且仅当t出现在R中但不出现在S中时输出
  - ✓对于包差： R $-_B$ S，输出t的次数是它在R中的次数减去其在S中出现的次数

基于排序的两趟扫描算法
(5)基于排序的连接运算?

战德臣 教授

$$R \bowtie S$$
$$R.Y = S.Y$$

- 第一趟：划分R和S的子表并进行子表排序，排序均基于Y属性排序。
- 第二趟：归并时注意是R的输入还是S的输入。R和S的两路输入之间进行连接检查并连接后输出。
- “排序-连接”算法，“归并-连接”算法，“排序-归并-连接”算法
- SORT-JOIN， MERGE-JOIN， SORT-MERGE-JOIN



基于散列的两趟扫描算法

战德臣

哈尔滨工业大学　教授.博士生导师

黑龙江省教学名师

教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent Computing for Enterprises & Services, Harbin Institute of Technology

基于散列的两趟扫描算法

(1)基本思想

大数据集上的操作 可以被转换为某个子集上的操作

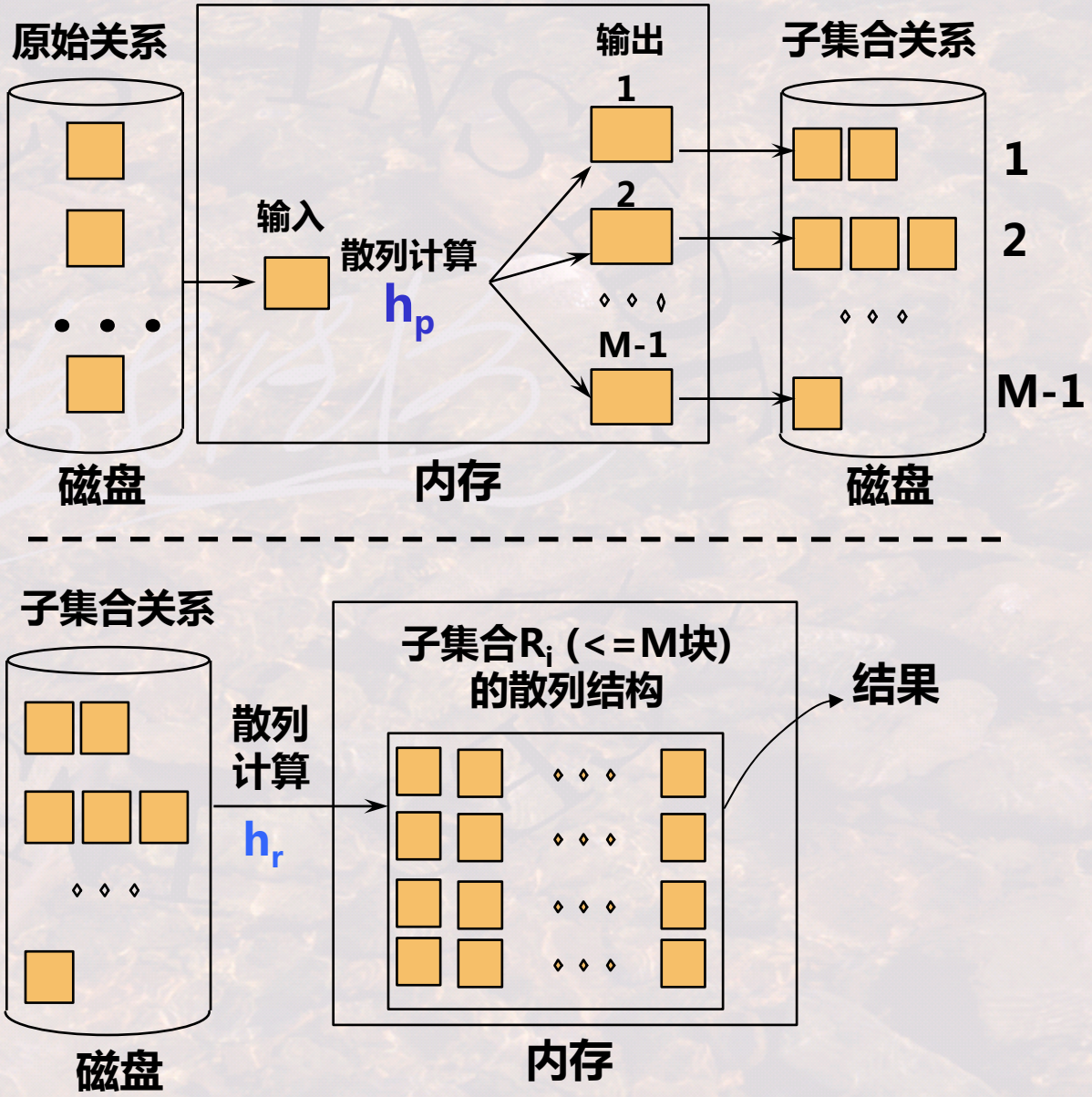
第一趟：散列子表。

用散列函数h_p将原始关系划分成M-1个子表，并存储

第二趟：处理每个子表

用另一散列函数h_r将子表读入内存并建立内存结构，进行不同操作的处理

散列函数随操作的不同而有不同的选择



基于散列的两趟扫描算法
(2)去重复操作?

战德臣 教授

$\delta(R)$ --DISTINCT

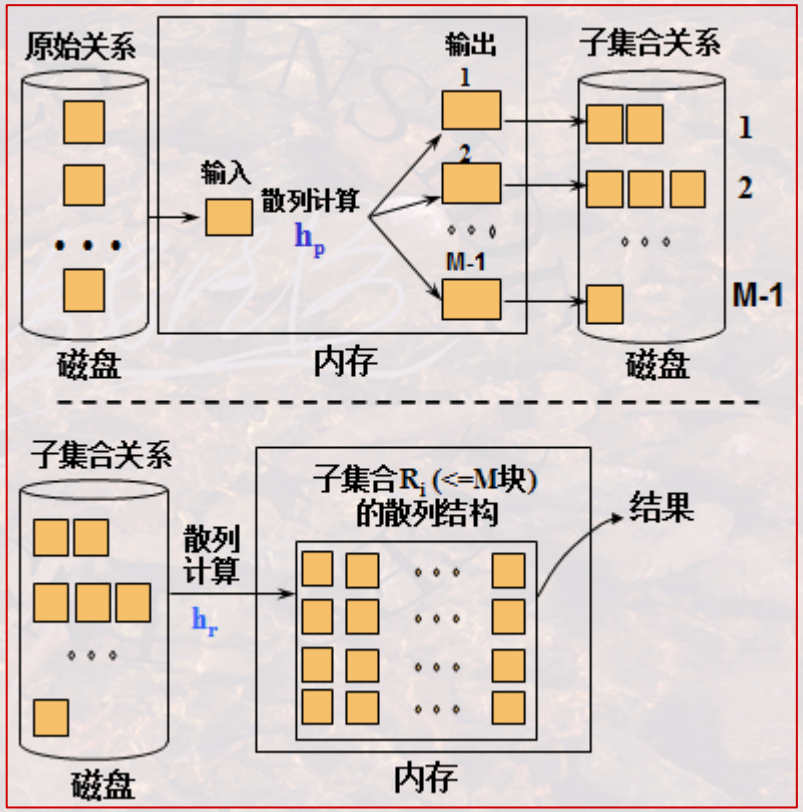
●第一趟：将原始关系通过h_p散列成M-1个子表，并进行存储；

●第二趟：处理每个子表。将每个子表读入内存，并用另一函数h_r形成散列数据结构，进行去重复操作。应选择不同的h_p和h_r，例如：如下是一种方案

●H_p= 计算元组部分属性的值 MOD M。

●H_r= 计算整个元组的值 MOD M。

●算法的复杂性，3B(R)---不考虑输出；4B(R)---考虑输出。



元组在子表上不重复，则在大关系中亦不重复

H_p将可能重复的元组散列到同一子表，h_r将可能重复的元组散列到同一内存块中

基于散列的两趟扫描算法
(3)分组聚集操作?

战德臣 教授

γ(R) --- GROUP BY 分组属性

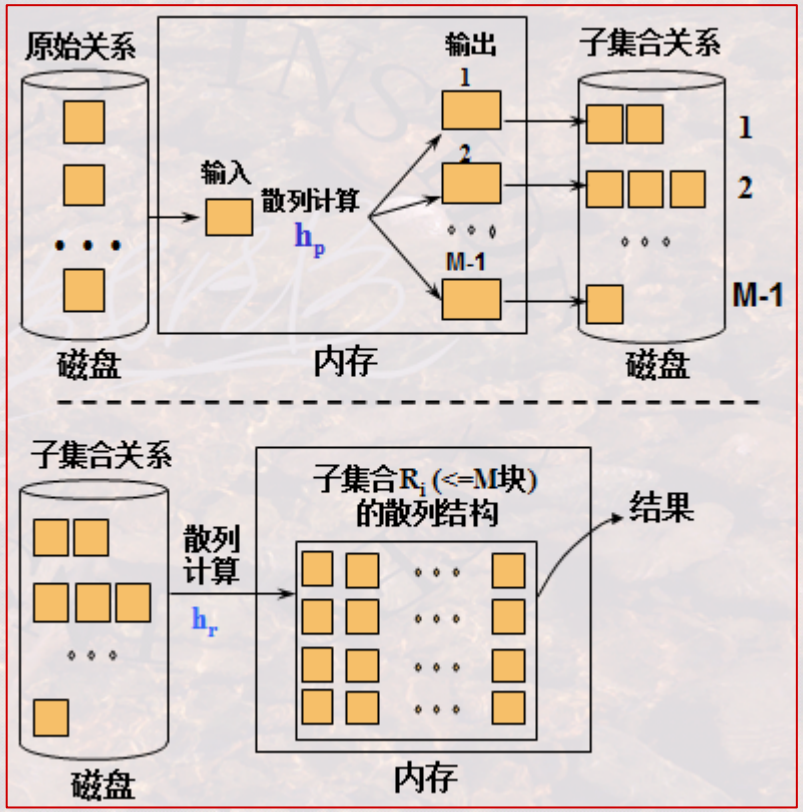
●第一趟：将原始关系通过h_p散列成M-1个子表，并进行存储；

●第二趟：处理每个子表。将每个子表读入内存，并用另一函数h_r形成散列数据结构，进行分组聚集操作。应选择不同的h_p和h_r，例如：如下是一种方案

●H_p= 计算“分组属性”的值 MOD M。

●H_r= 以“分组属性”的二进制位串重新计算值，然后 MOD M。

●算法的复杂性，3B(R)---不考虑输出；4B(R)---考虑输出。



同一分组的所有记录应在同一个子表中：H_p。同一子表中同一分组的所有记录应在同一内存块中：H_r。 H_p和H_r不能相同，但都以分组属性为基础。

基于散列的两趟扫描算法
(4)基于散列的并、交和差操作?

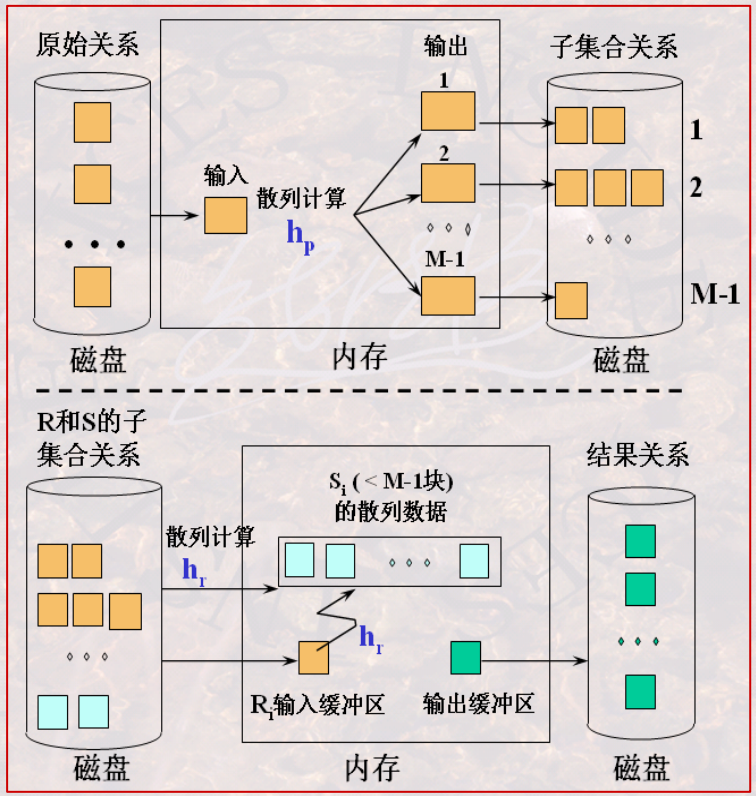
战德臣 教授

集合上的操作：$\cup_S$ ，$\cap_S$，$-_S$　包上的操作：$\cup_B$ ，$\cap_B$，$-_B$

- 使用相同散列函数散列两个操作对象R和S，形成$R_1,\ldots,R_M$和$S_1,\ldots,S_M$，

  R operator S = $\cup_{i=1,\ldots,M}(R_i$ operator $S_i)$

- $\cup_B$ 无需两趟，直接两个关系合并即可；
- $\cup_S$ 需两趟，需要去重复。第一趟：以相同散列函数散列R和S形成M-2个子表R_i,S_i；第二趟：将S_i再整体散列读入到内存中，再依次处理R_i的每一块。如判断在R_i,S_i都出现元组t,则仅输出t的一个副本，否则输出S_i和R_i。
- $\cap_S$和 $\cap_B$：可采取相似方法处理(略)。
- $-_S$和 $-_B$：可采取相似方法处理(略)。



基于散列的两趟扫描算法
(5)基于散列的连接操作?

战德臣 教授

$$R \underset{R.Y = S.Y}{\bowtie} S$$

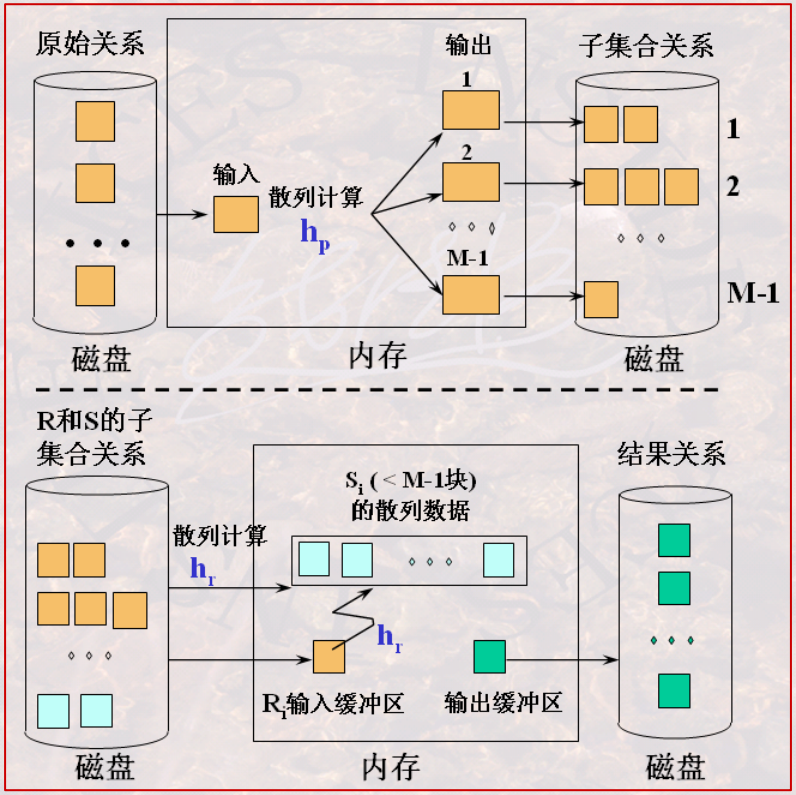
●以连接属性Y作散列关键字，设计散列函数。

●第一趟：使用相同散列函数散列两个操作对象R和S，形成$R_1,\ldots,R_M$和$S_1,\ldots,S_M$，

$$R \underset{R.Y = S.Y}{\bowtie} S = \cup_{i=1,\ldots,M}(R_i \underset{R_i.Y = S_i.Y}{\bowtie} S_i)$$

●第二趟：将S_i再整体散列读入到内存中，再依次处理R_i的每一块。进行连接。

● “散列连接”算法



回顾本讲学了什么?

战德臣

哈尔滨工业大学　教授.博士生导师

黑龙江省教学名师

教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent Computing for Enterprises & Services, Harbin Institute of Technology

回顾本讲学习了什么?

ICES 战德臣 教授

查询实现算法的基本思维

两趟算法

第一趟

第二趟

大数据集划分多个子集并使子集具有某种特性(排序/散列)

如何由多个子集经处理得到最终结果

一次单一元组的一元运算

整个关系上的一元运算

整个关系上的二元运算

并、交、差、选择、投影

排序、去重复、分组计算

两阶段多路归并排序

外存数据结构
+
内存数据结构

基于排序的两趟算法

基于散列的两趟算法