实验五　查询处理算法的模拟实现

2022秋

本学期实验总体安排

本学期实验课程共 16 个学时，5 个实验项目，总成绩为 30 分。

实验项目	实验一	实验二	实验三	实验四		实验五
学时	2	2	2	4	2	4
实验内容	MySQL及SQL的使用	高级SQL语言的使用	openGauss的AI特性实验	一个小型系统的设计与实现		查询处理算法的模拟实现
分数	4	4	4	10		8

目录

实验目的

- 理解索引、散列的作用；
- 掌握关系选择、投影、连接、集合的交、并、差等操作的实现算法；
- 加深对算法I/O复杂性的理解。

实验内容

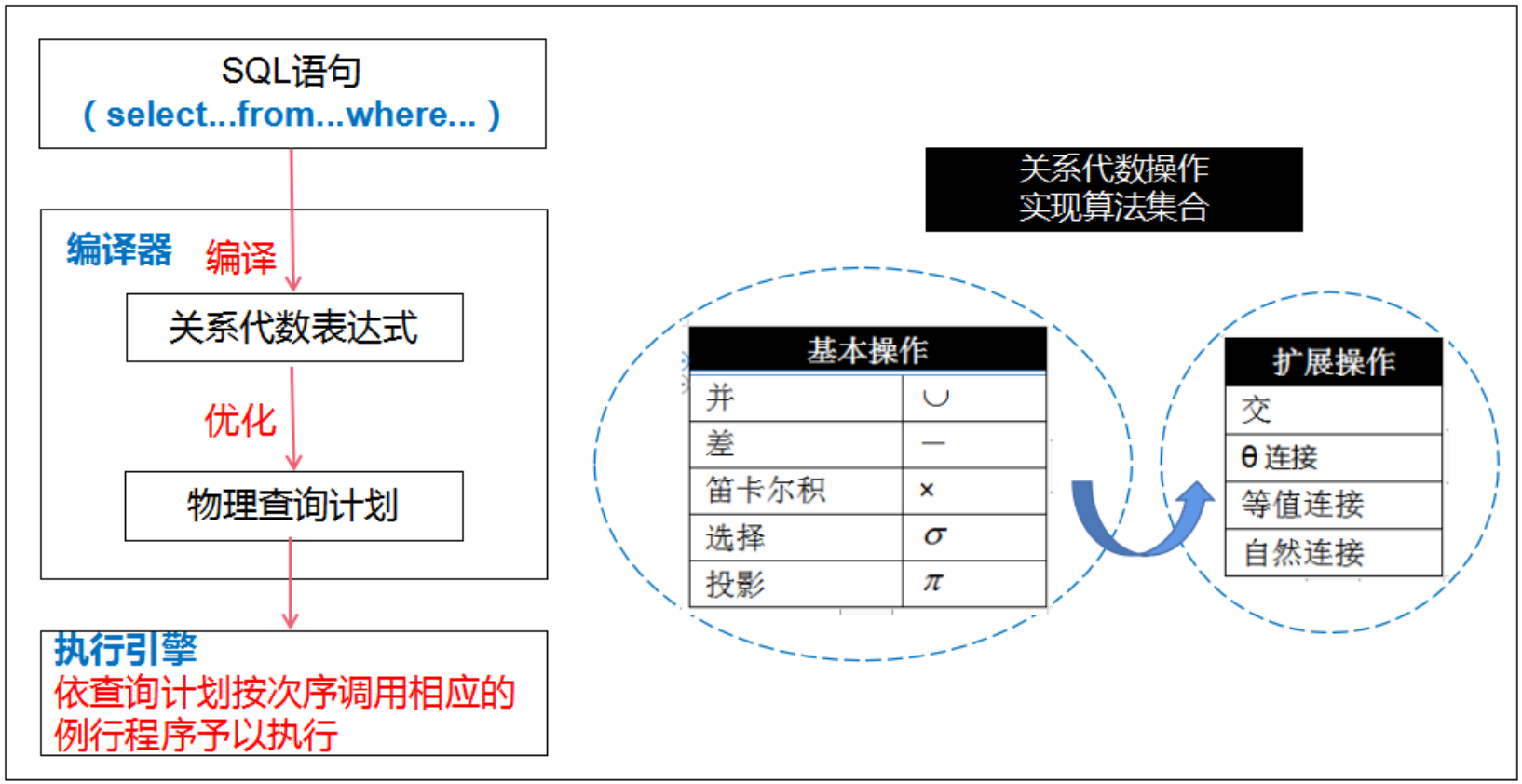
1. 基于ExtMem程序库，实现关系选择、连接操作算法

2. 实现集合并、交、差操作算法

要求：使用有限内存（Buffer）实现上述算法，不可定义大数组装载数据和保存中间结果。

实验原理

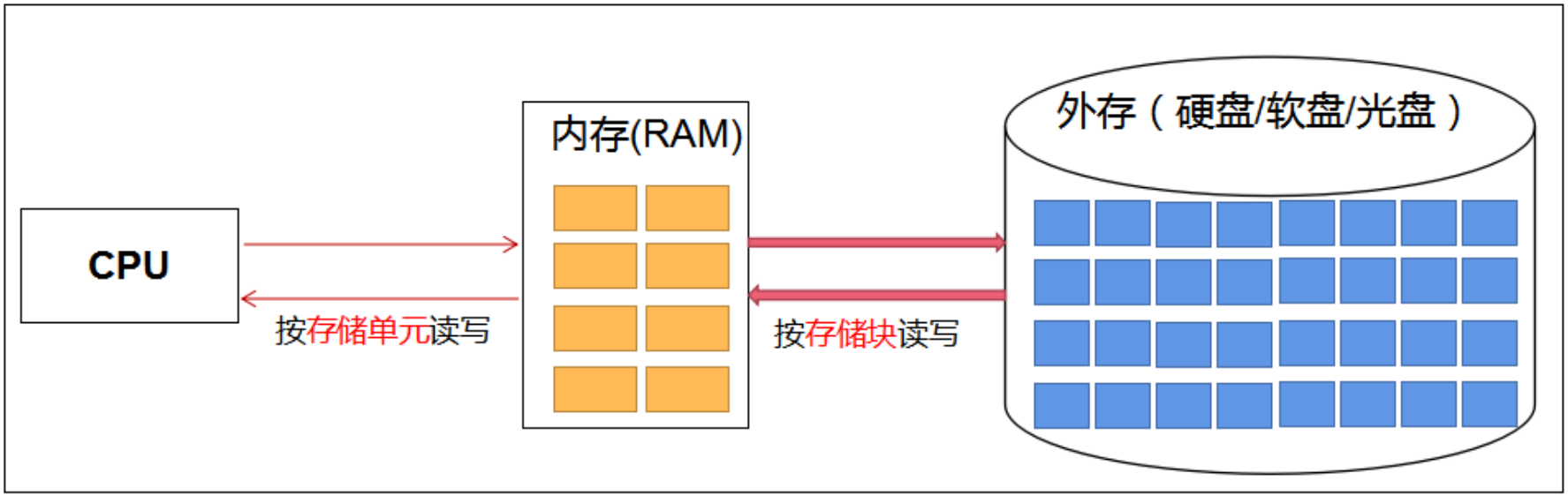
DBMS查询实现的基本思想：



实验原理

CPU、内存和外存的关系

- CPU与内存直接交换信息，按存储单元（存储字）进行访问
- 外存按存储块进行访问，其信息需现装入内存，才能被CPU处理



实验原理

索引

定义在存储表基础上，有助于无需检查所有记录而快速定位所需记录的一种辅助存储结构。

主文件

essn	ename	address	salary	superssn	dno
00	白玉芬	广州市天河区珠江新城花城大道83号	1000	24	100
01	仓春莲	广东省广州市石牌西路68号	1100	20	100
02	仓红	广州市番禺区市桥清河东路41号	2000	14	011
03	陈超云	佛山市汾江西路13号	3000	02	011
04	陈高	佛山市南海区桂城天佑三路1号大院	4000	06	001
05	陈国祥	佛山市顺德区容奇大道东23号	3500	09	000
06	陈宏柳	肇庆市端州七路13号	2800		001
07	陈金娣	广州市石牌西路68号	4500	14	011
08	陈丽丽	清远市北江三路海关大楼6楼	2100	24	100
09	陈平	深圳市南山区西丽大学城	3000		000
10	陈向东	云浮市建设南路60号	2500	09	000
11	陈晓冬	广州市花都区建设北路79号	3500	01	100
12	陈小荣	广州市机场路561号	3600		010
13	陈秀芬	韶关市环园东路1号	2600	24	100
14	陈艳华	佛山市禅城区汾江西路13号	4600		011
15	陈兆国	广州市天河区珠江新城华利路61号	2700	12	010
16	成秀山	从化市街口街关前路1栋	1900	01	100
17	仇腊梅	广州市站南路4号流花邮政大院	1800	01	100
18	戴金辉	清远市北江三路	3300	01	100
19	邓海燕	广州市番禺区清河东路43号	2000	12	010
20	翟苗	广州市南沙区南沙街进港大道1	4000		100
21	丁德明	广州市白云区	2900	07	011
22	丁素琴	广州市萝岗区开创大道北1168号	2500	24	100
23	董荣柱	河源市河源大道北229号	1500	19	010
24	张红	肇庆市端州七路13号	2000	20	100

主索引文件

essn	
00	
08	
16	
24	

索引字段　　块指针

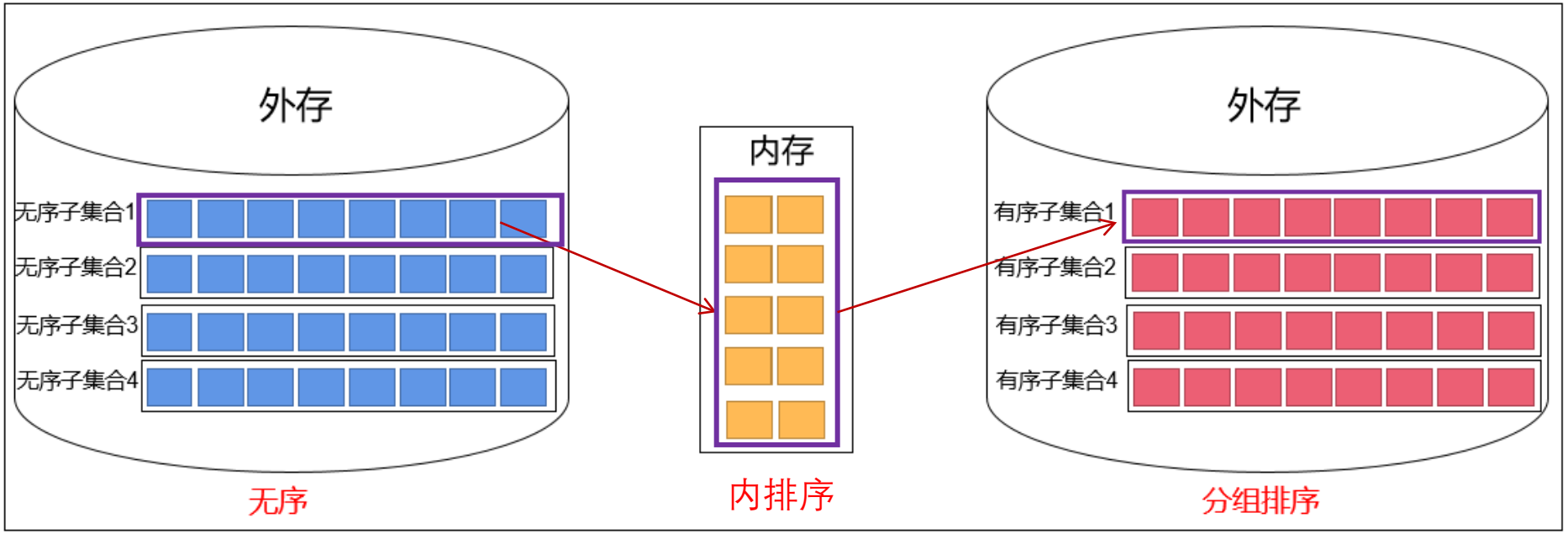
实验原理

两阶段多路归并排序算法（TPMMS）

待排序数据不能一次装入内存，需将数据分批装入分批处理。

基本思想：

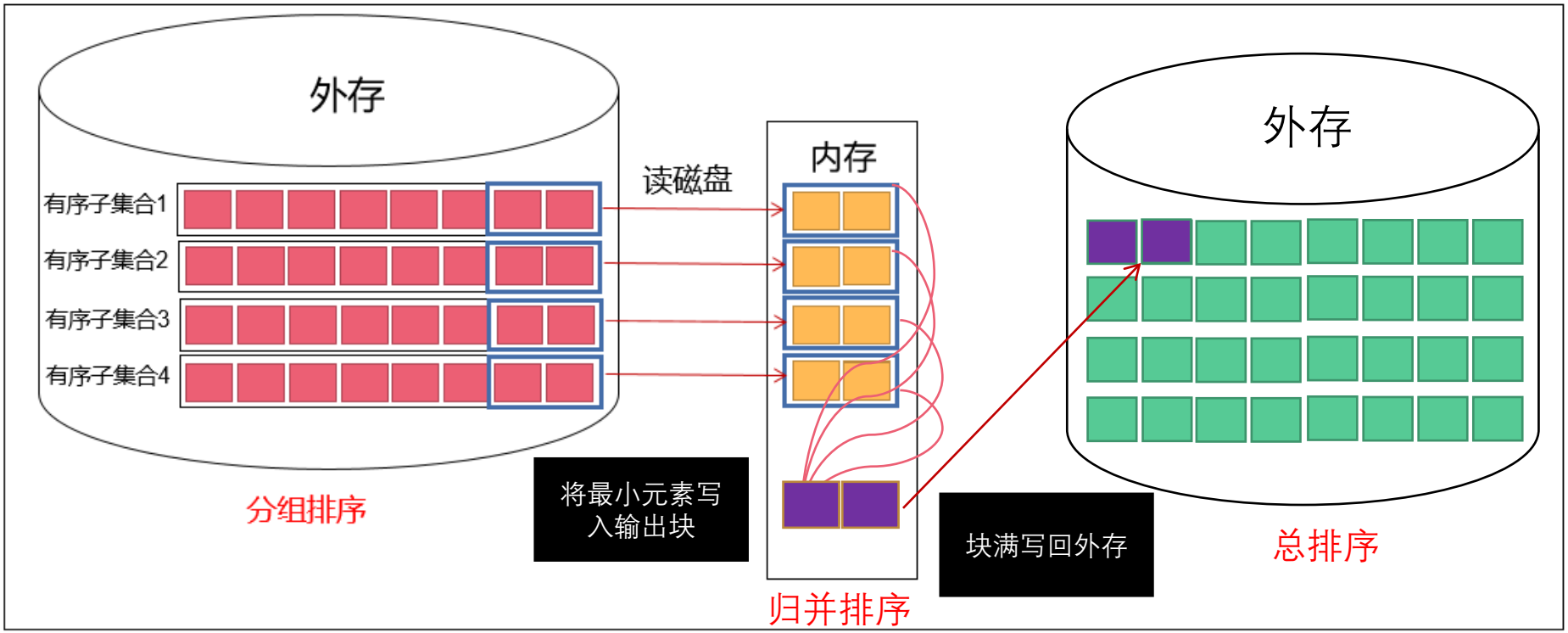
(一) 划分子集并子集排序



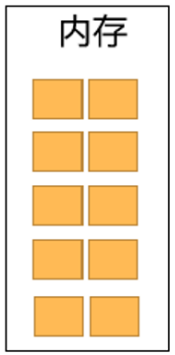
实验原理

两阶段多路归并排序算法（TPMMS）

(二) 各子集间归并排序



实验原理



两阶段多路归并排序算法（TPMMS）

(二) 各子集间归并排序

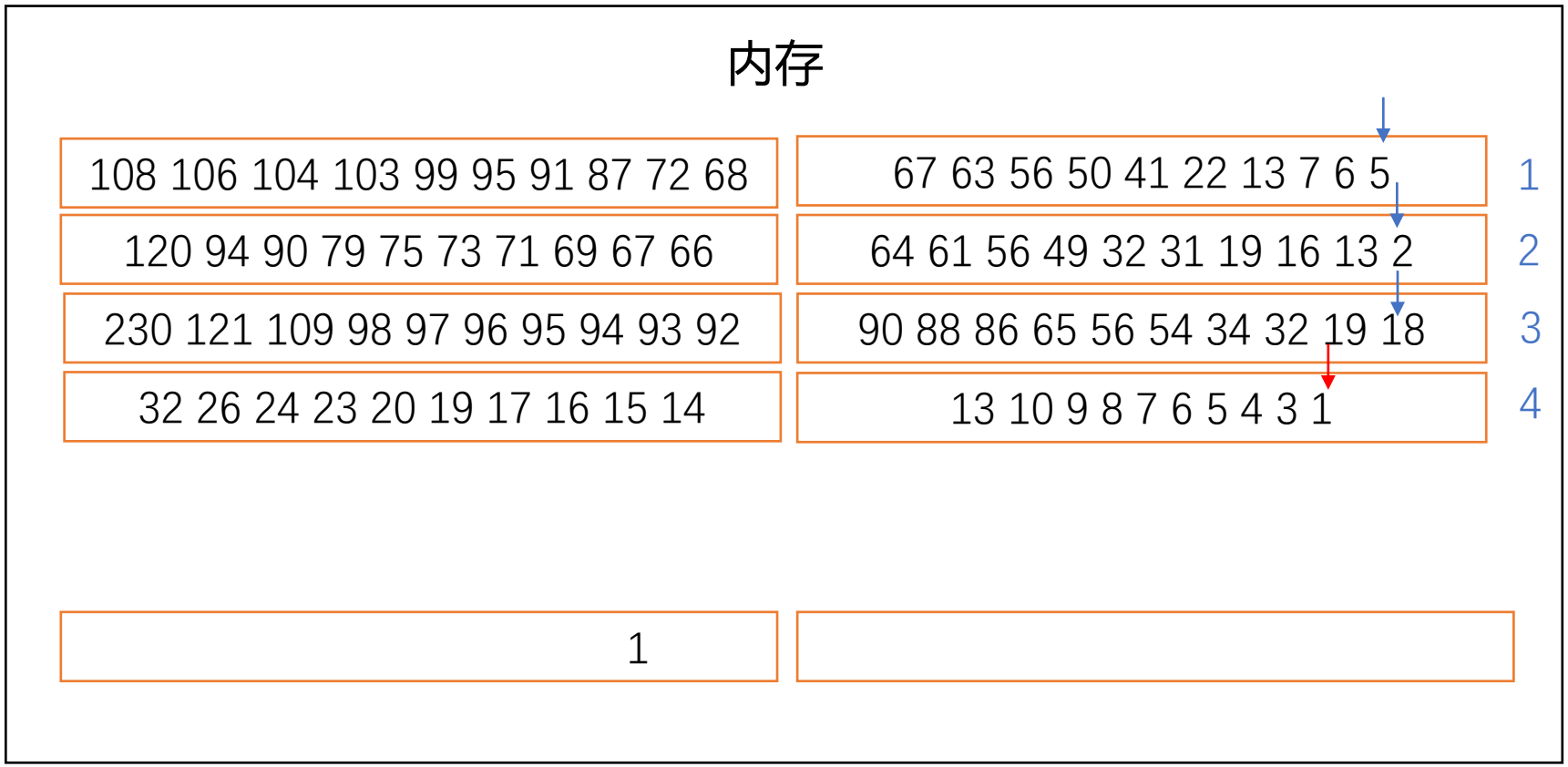
内存

108 106 104 103 99 95 91 87 72 68	67 63 56 50 41 22 13 7 6 5	1
120 94 90 79 75 73 71 69 67 66	64 61 56 49 32 31 19 16 13 2	2
230 121 109 98 97 96 95 94 93 92	90 88 86 65 56 54 34 32 19 18	3
32 26 24 23 20 19 17 16 15 14	13 10 9 8 7 6 5 4 3 1	4

实验原理

两阶段多路归并排序算法（TPMMS）

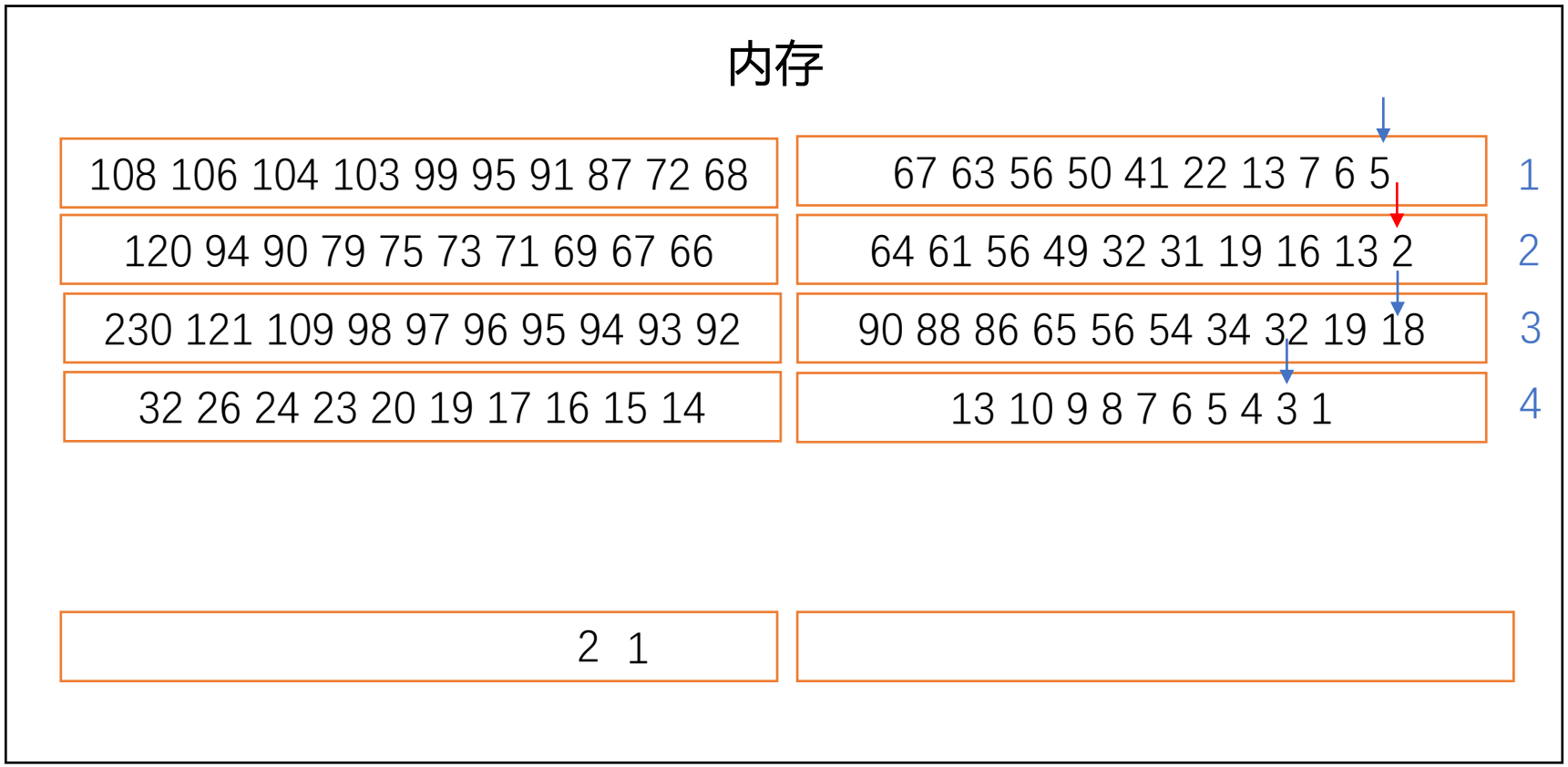
(二) 各子集间归并排序



实验原理

两阶段多路归并排序算法（TPMMS）

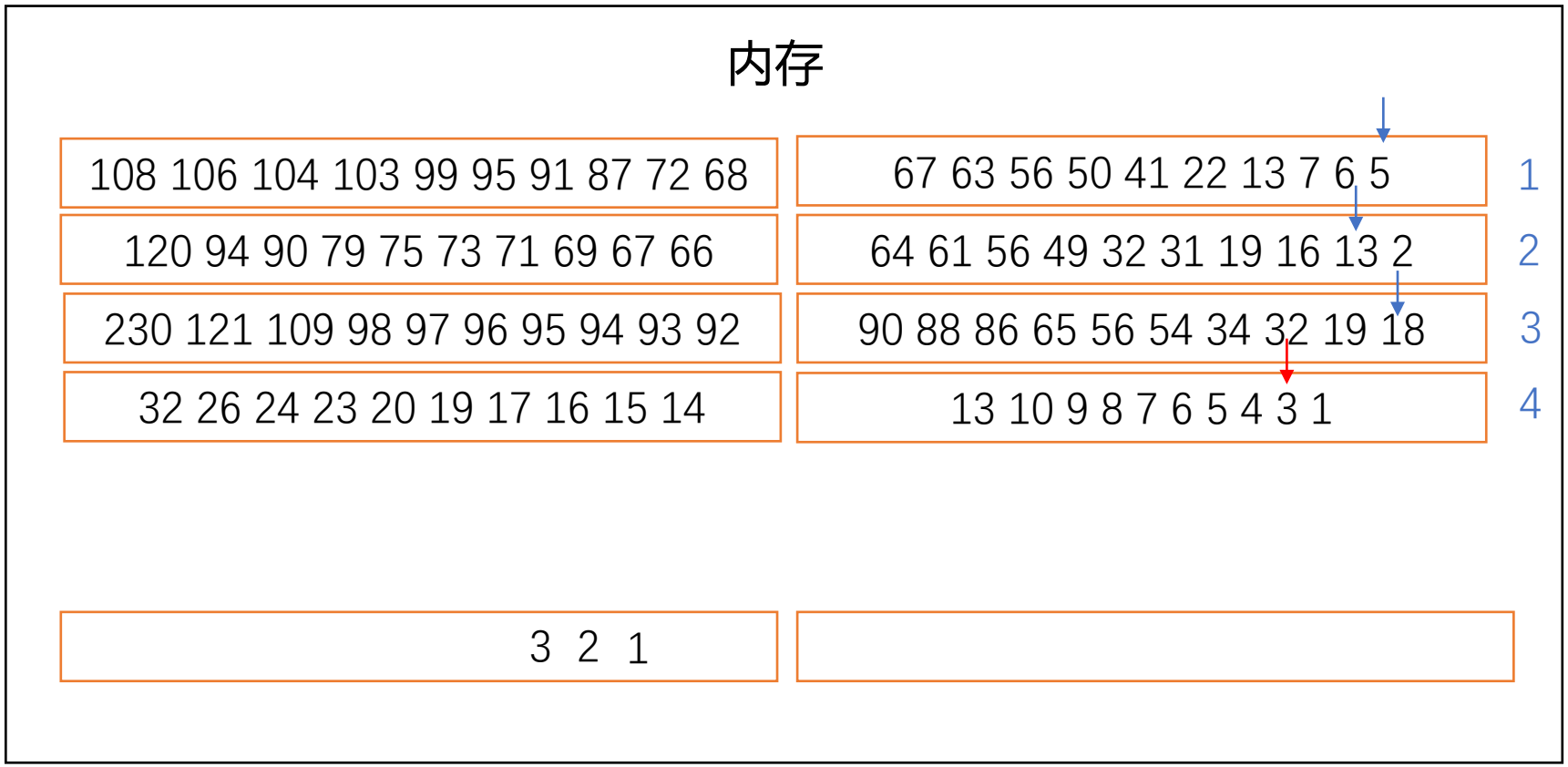
(二) 各子集间归并排序



实验原理

两阶段多路归并排序算法（TPMMS）

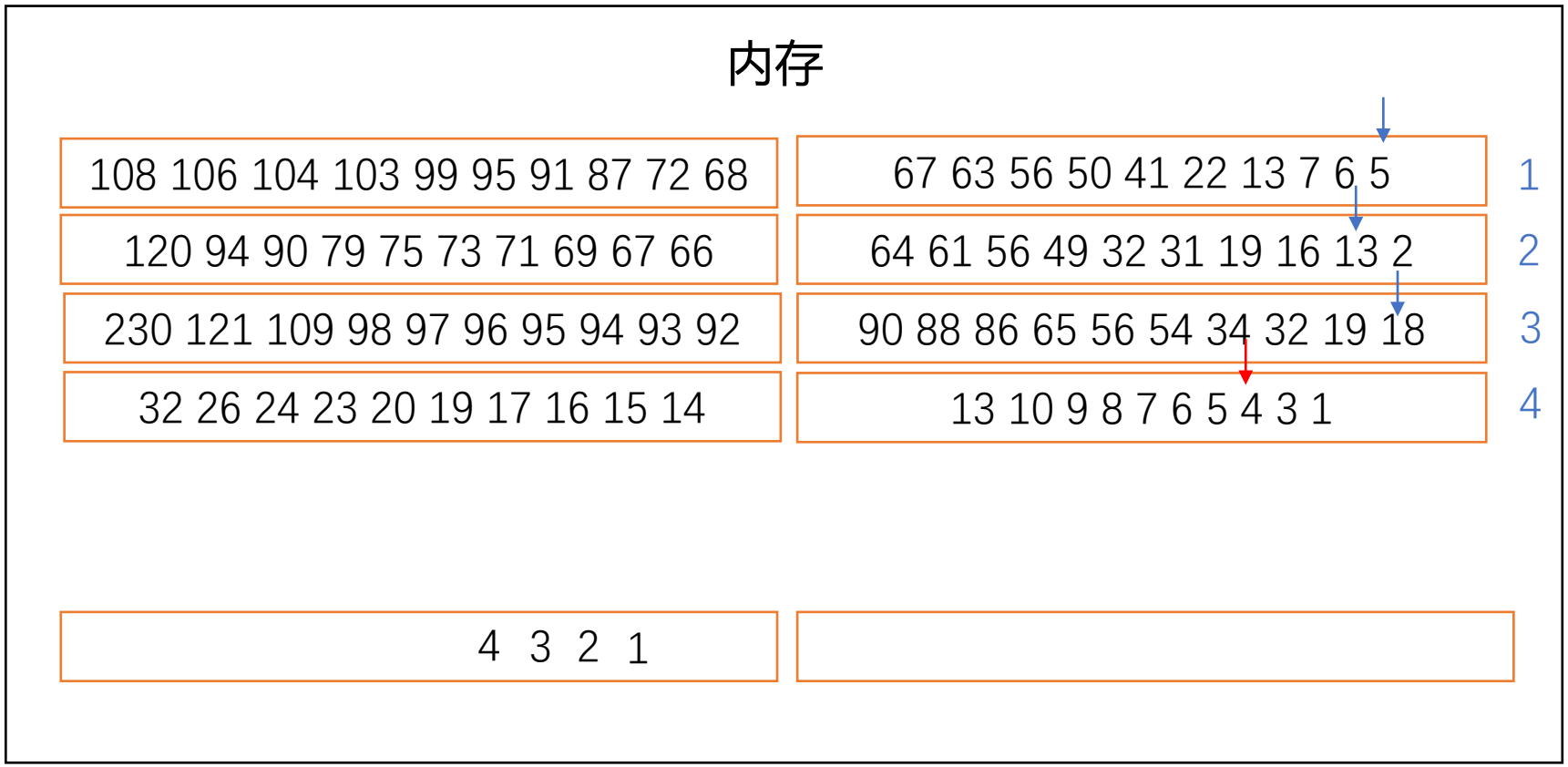
(二) 各子集间归并排序



实验原理

两阶段多路归并排序算法（TPMMS）

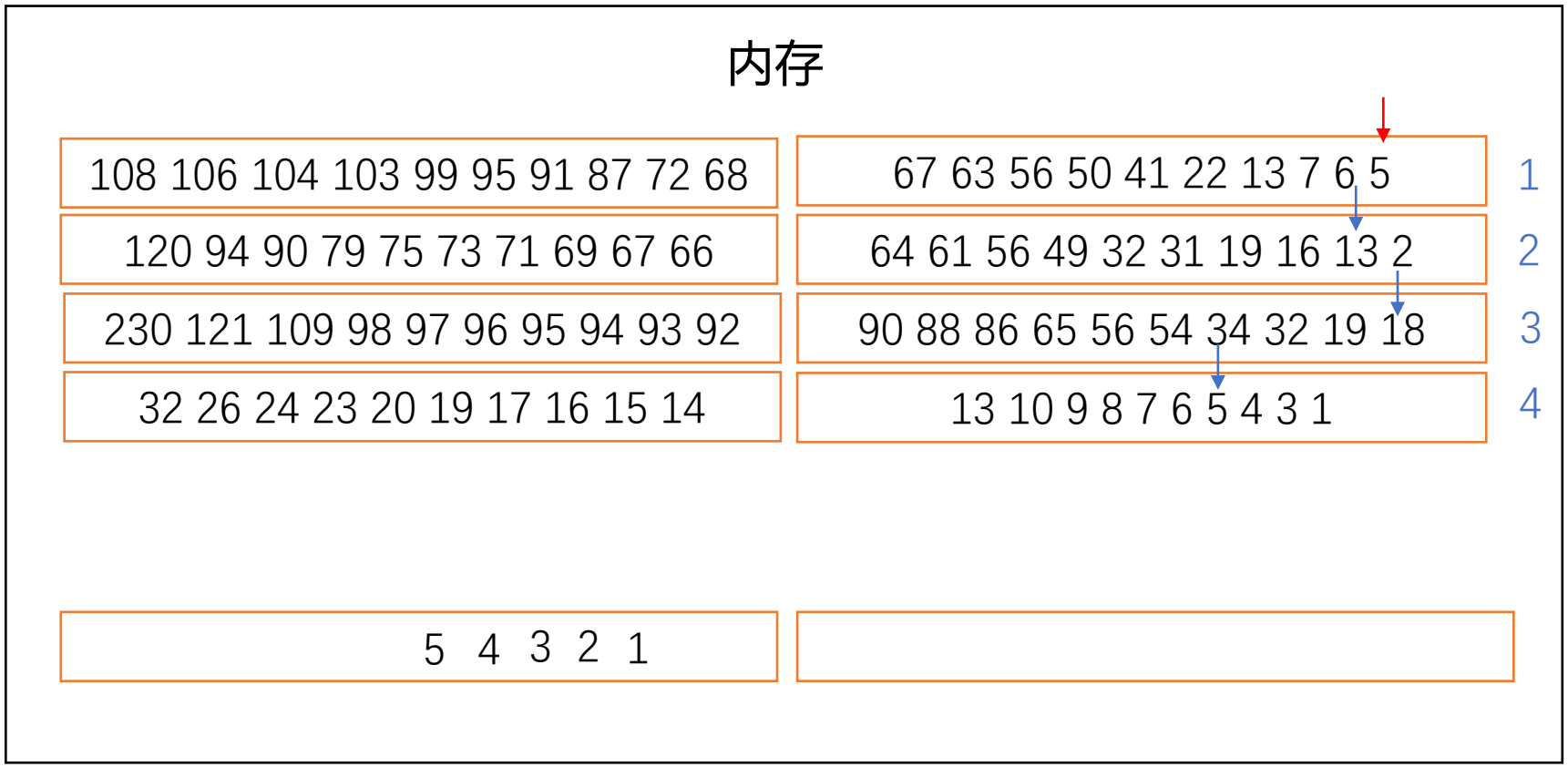
(二) 各子集间归并排序



实验原理

两阶段多路归并排序算法（TPMMS）

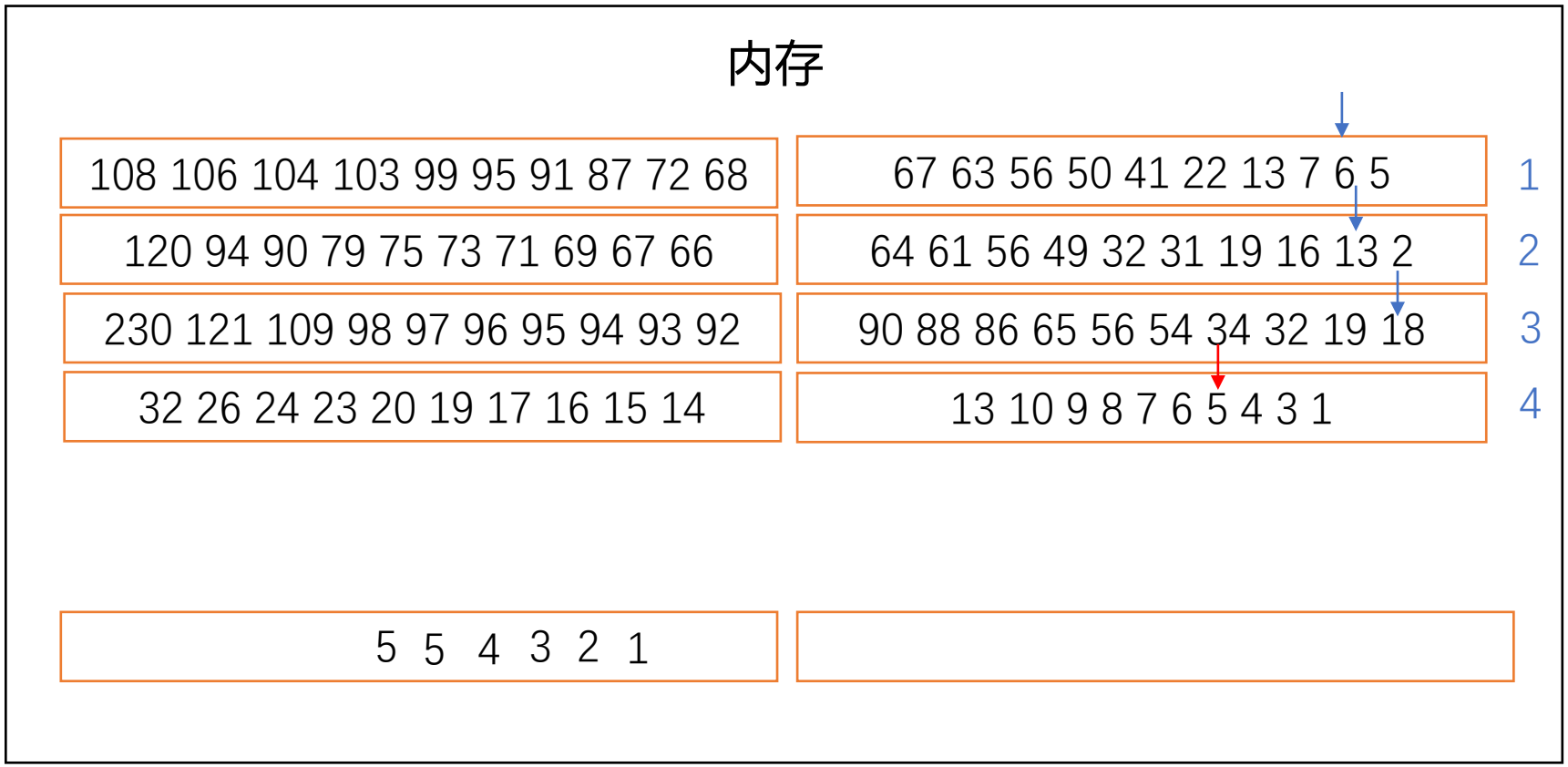
(二) 各子集间归并排序



实验原理

两阶段多路归并排序算法（TPMMS）

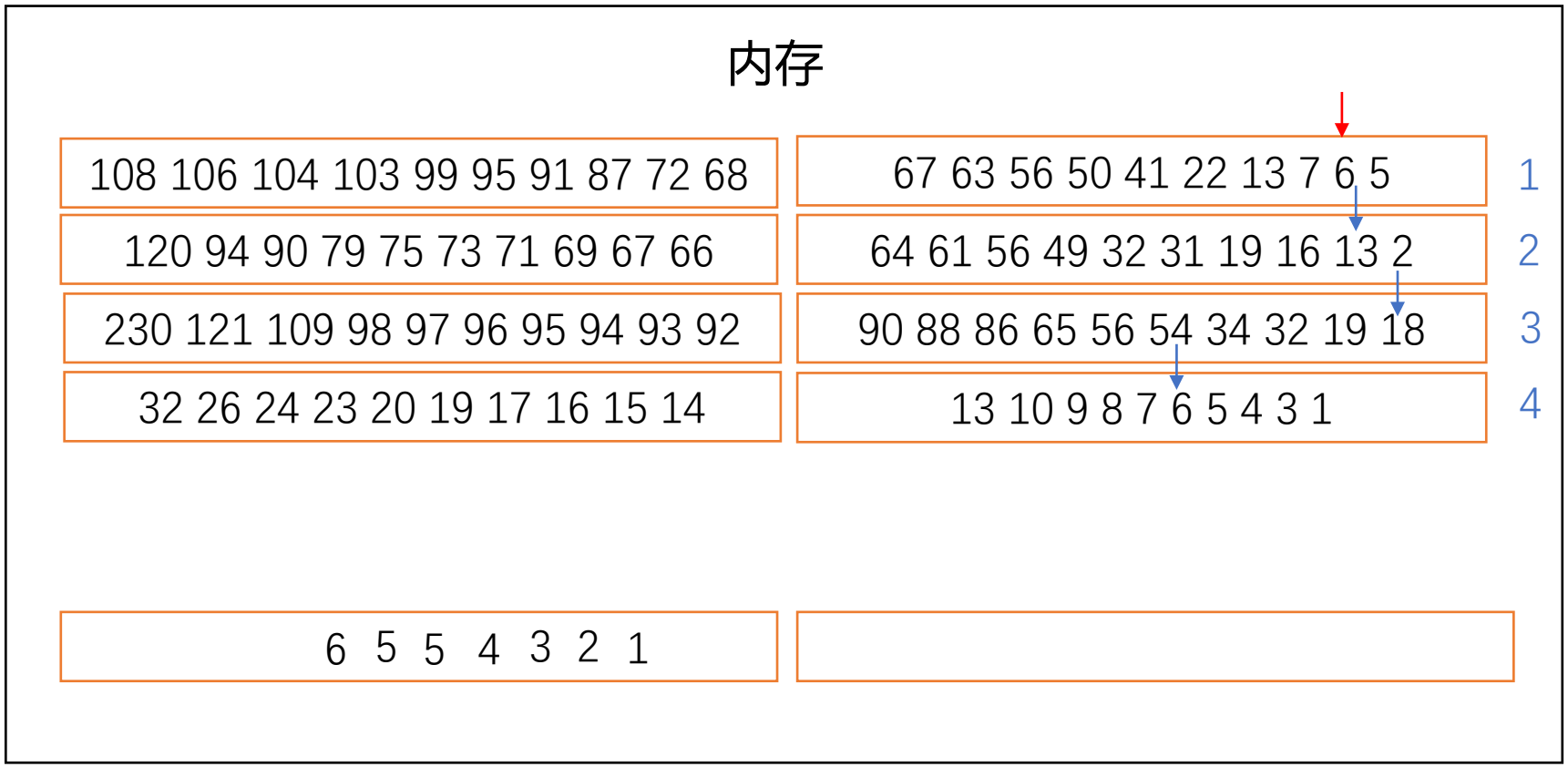
(二) 各子集间归并排序



实验原理

两阶段多路归并排序算法（TPMMS）

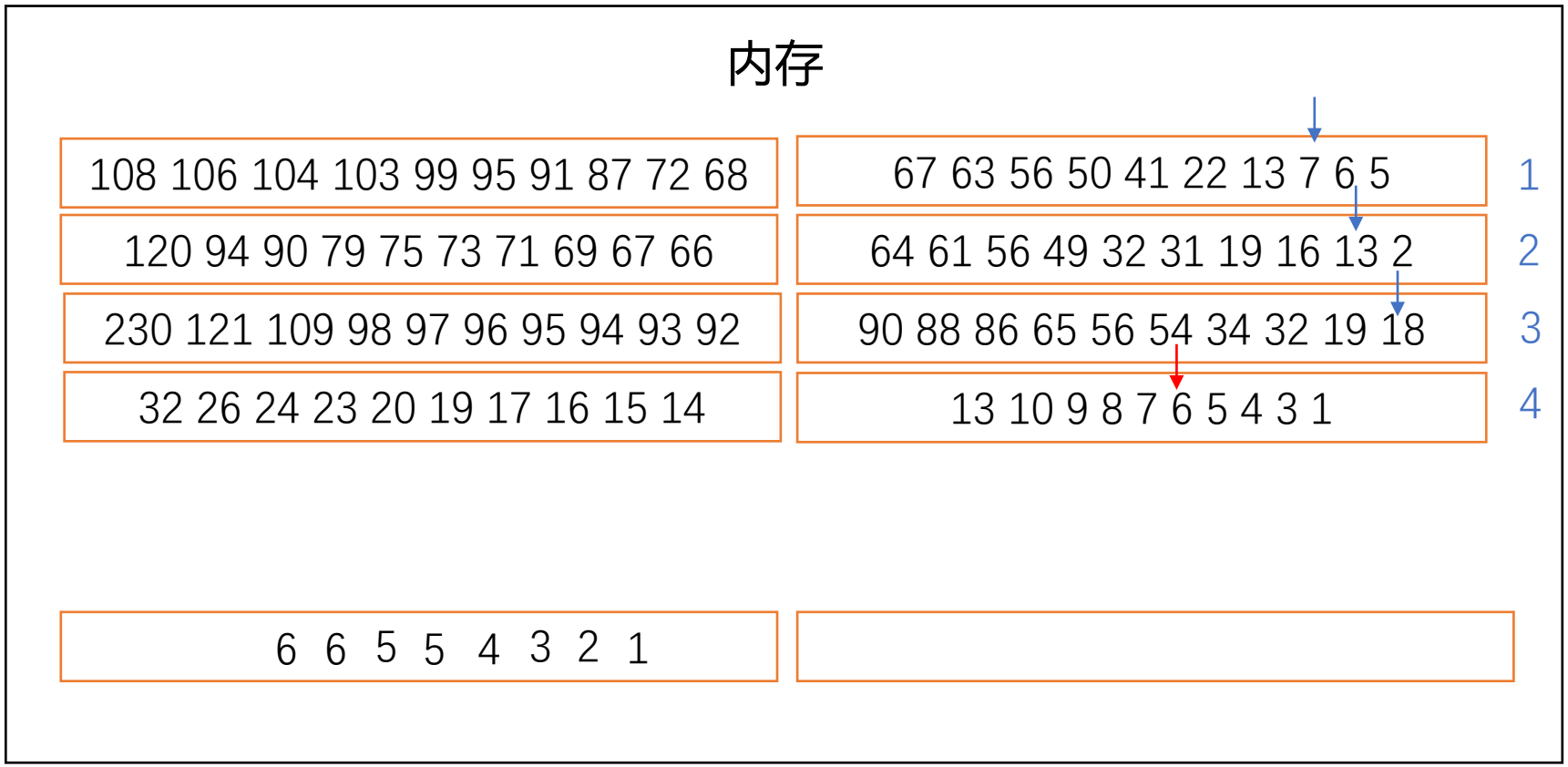
(二) 各子集间归并排序



实验原理

两阶段多路归并排序算法（TPMMS）

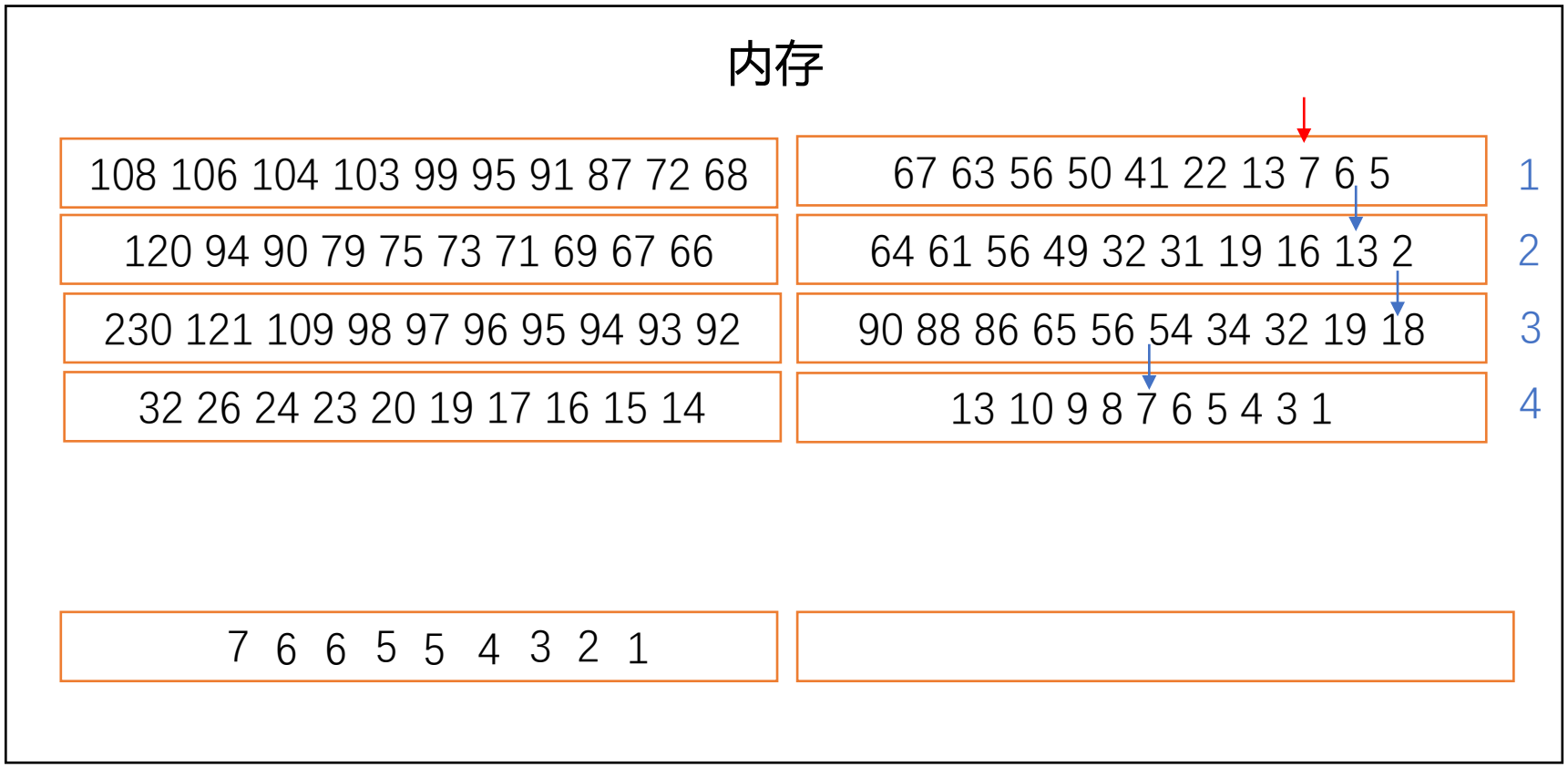
(二) 各子集间归并排序



实验原理

两阶段多路归并排序算法（TPMMS）

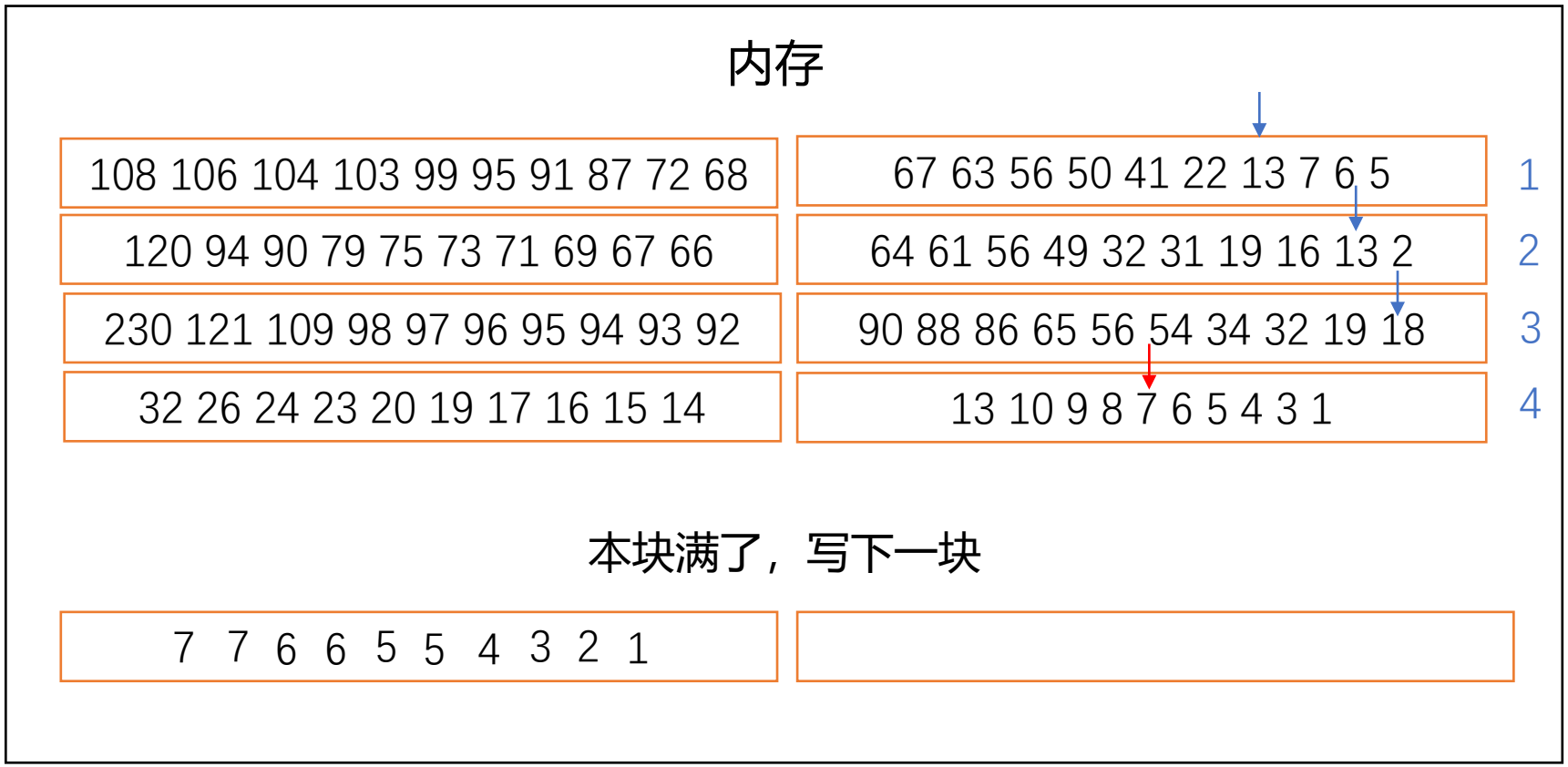
(二) 各子集间归并排序



实验原理

两阶段多路归并排序算法（TPMMS）

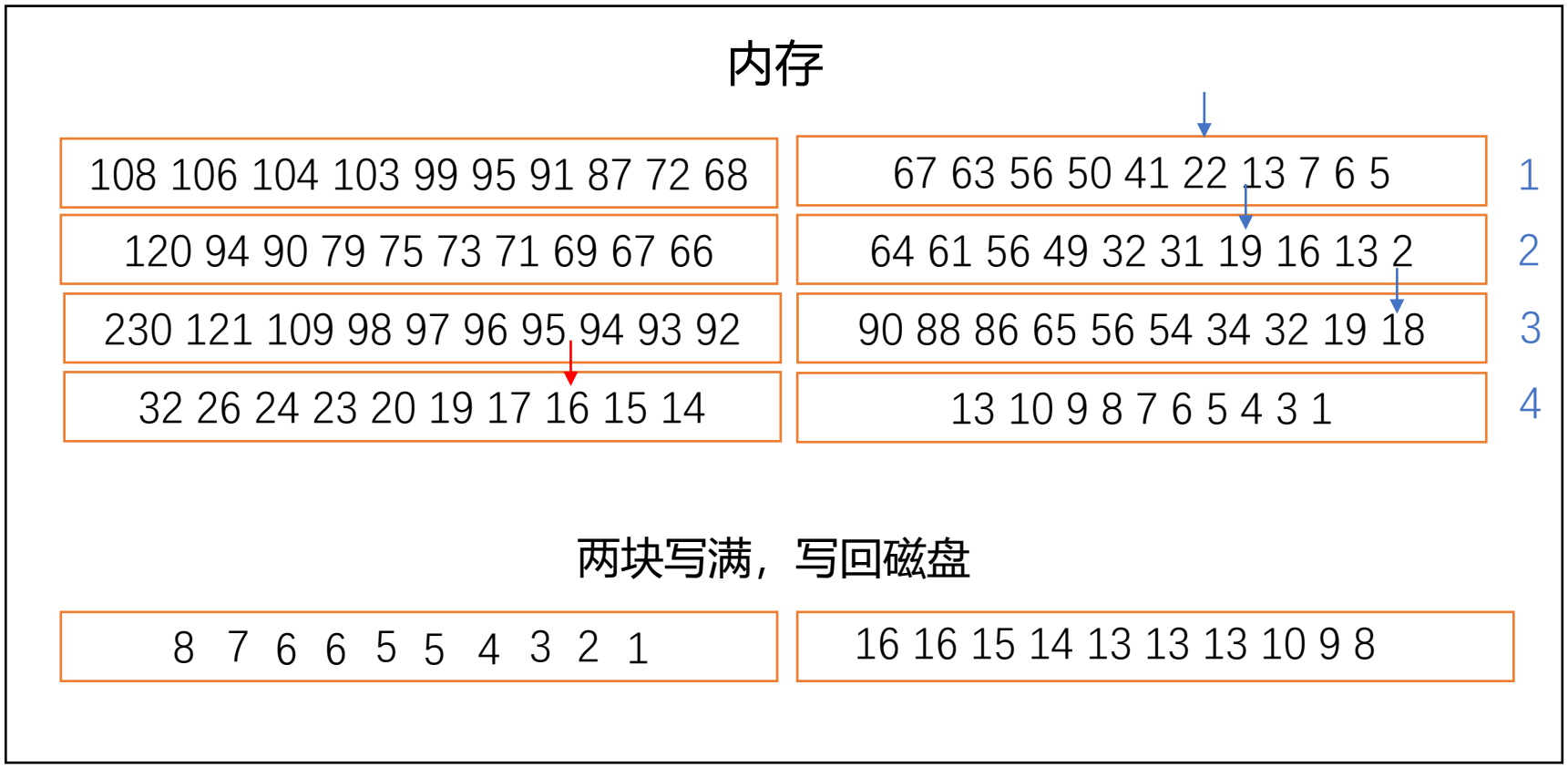
(二) 各子集间归并排序



实验原理

两阶段多路归并排序算法（TPMMS）

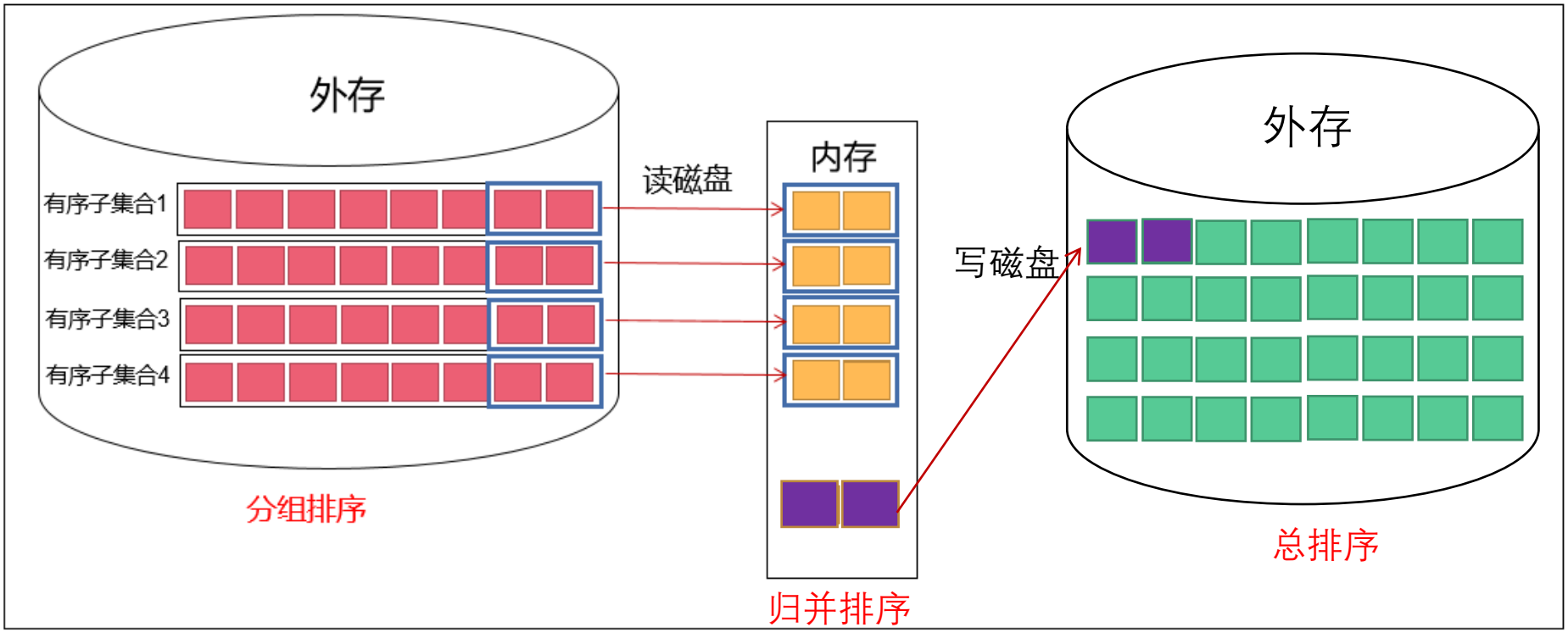
(二) 各子集间归并排序



实验原理

两阶段多路归并排序算法（TPMMS）

(二) 各子集间归并排序



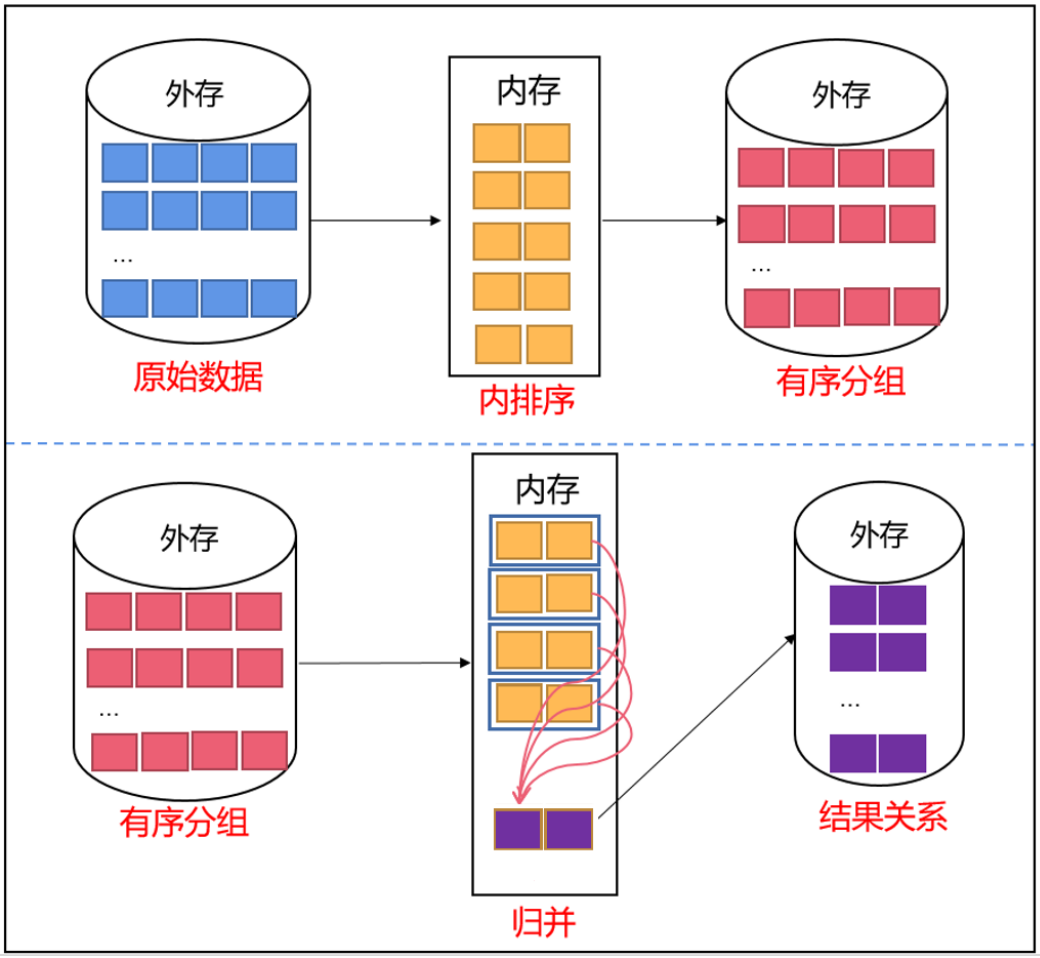
实验原理

基于排序的两趟扫描算法

第一趟：划分子表并进行子表排序

第二趟：归并阶段

- 关系一元操作
  - 去重复
  - 排序
  - 分组聚集
- 关系二元操作
  - 连接
  - 集合的并、交、差

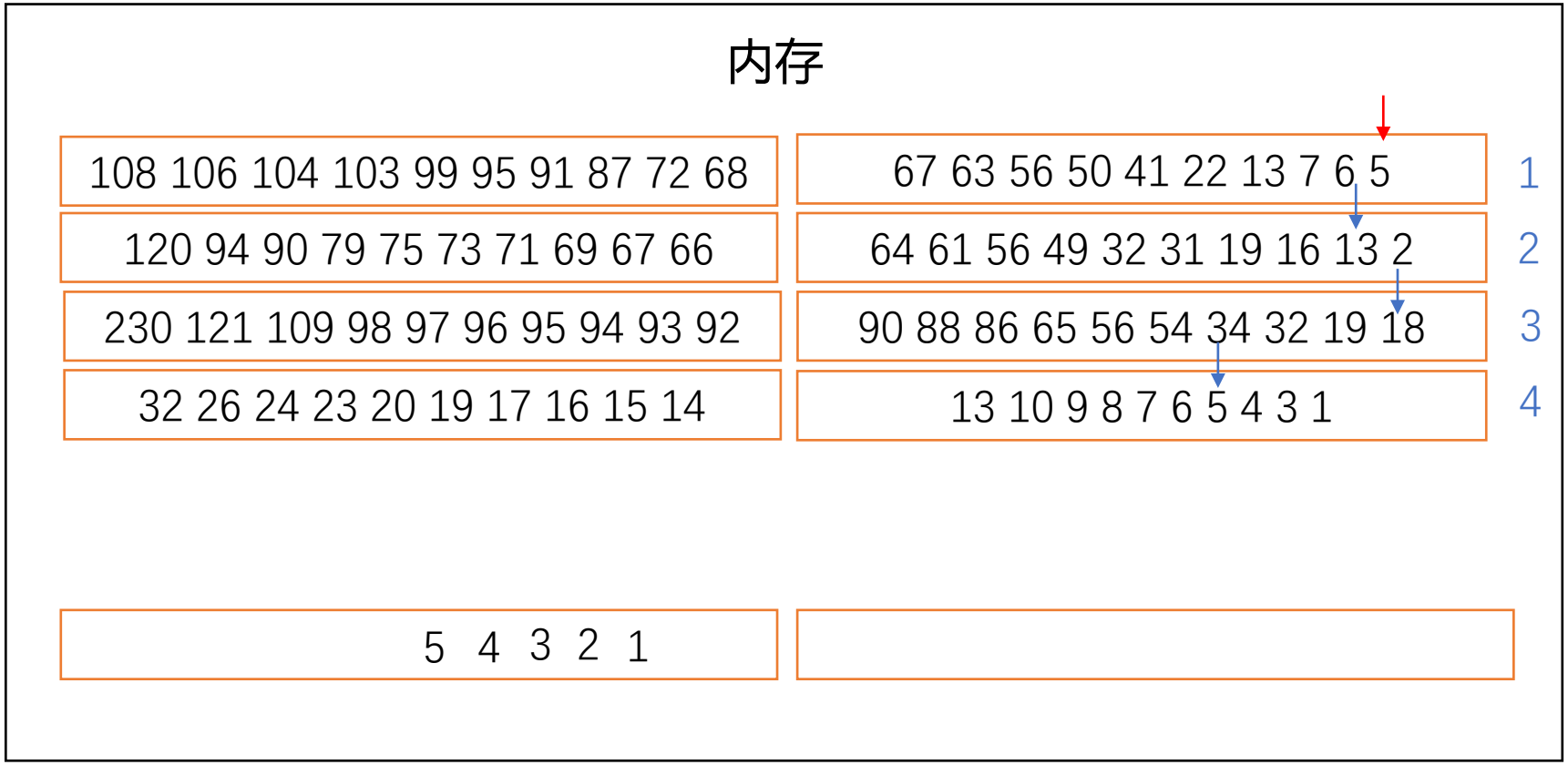


实验原理

基于排序的两趟扫描算法

第二趟：归并阶段

- 关系一元操作：去重复

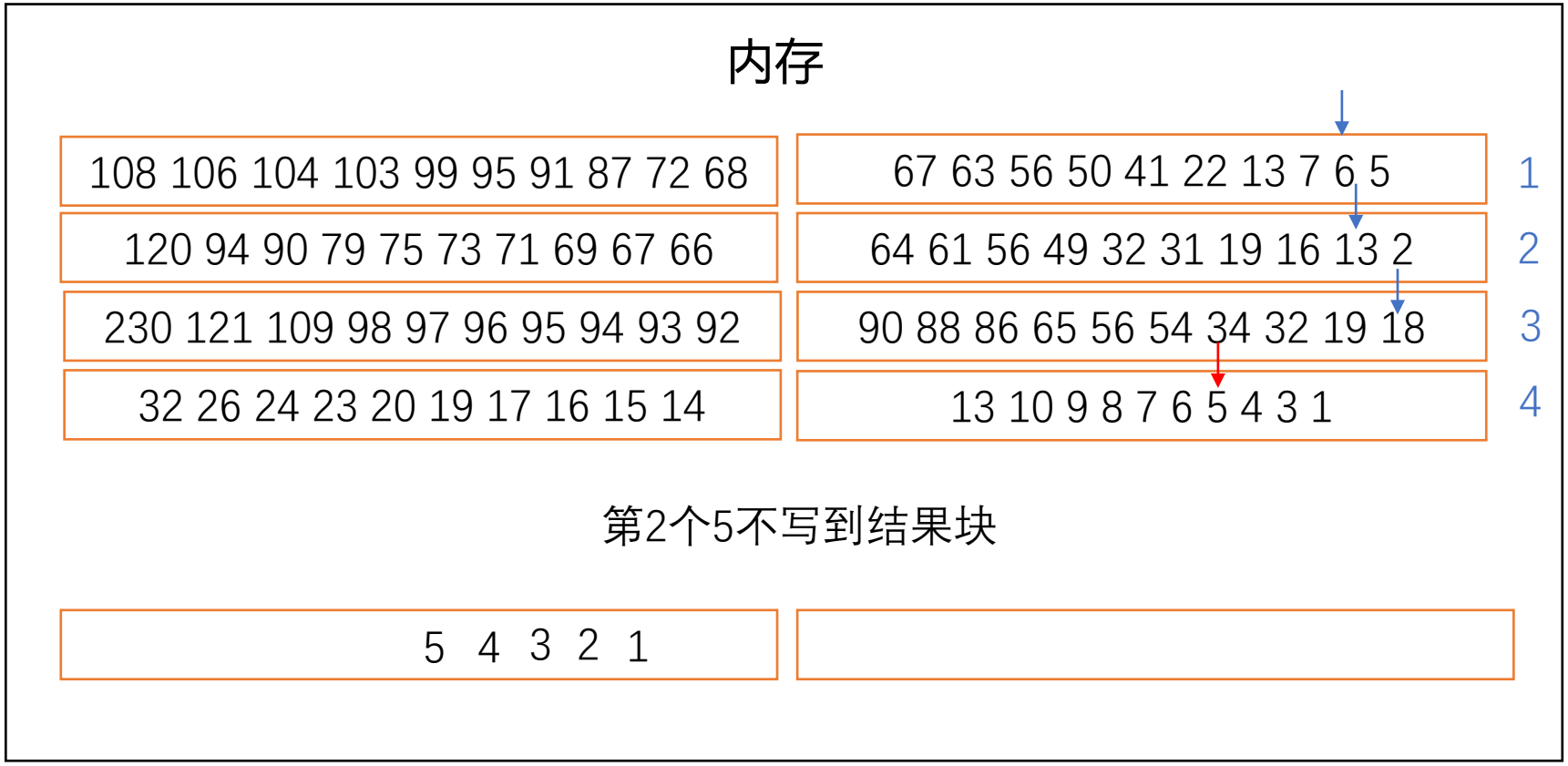


实验原理

基于排序的两趟扫描算法

第二趟：归并阶段

- 关系一元操作：去重复

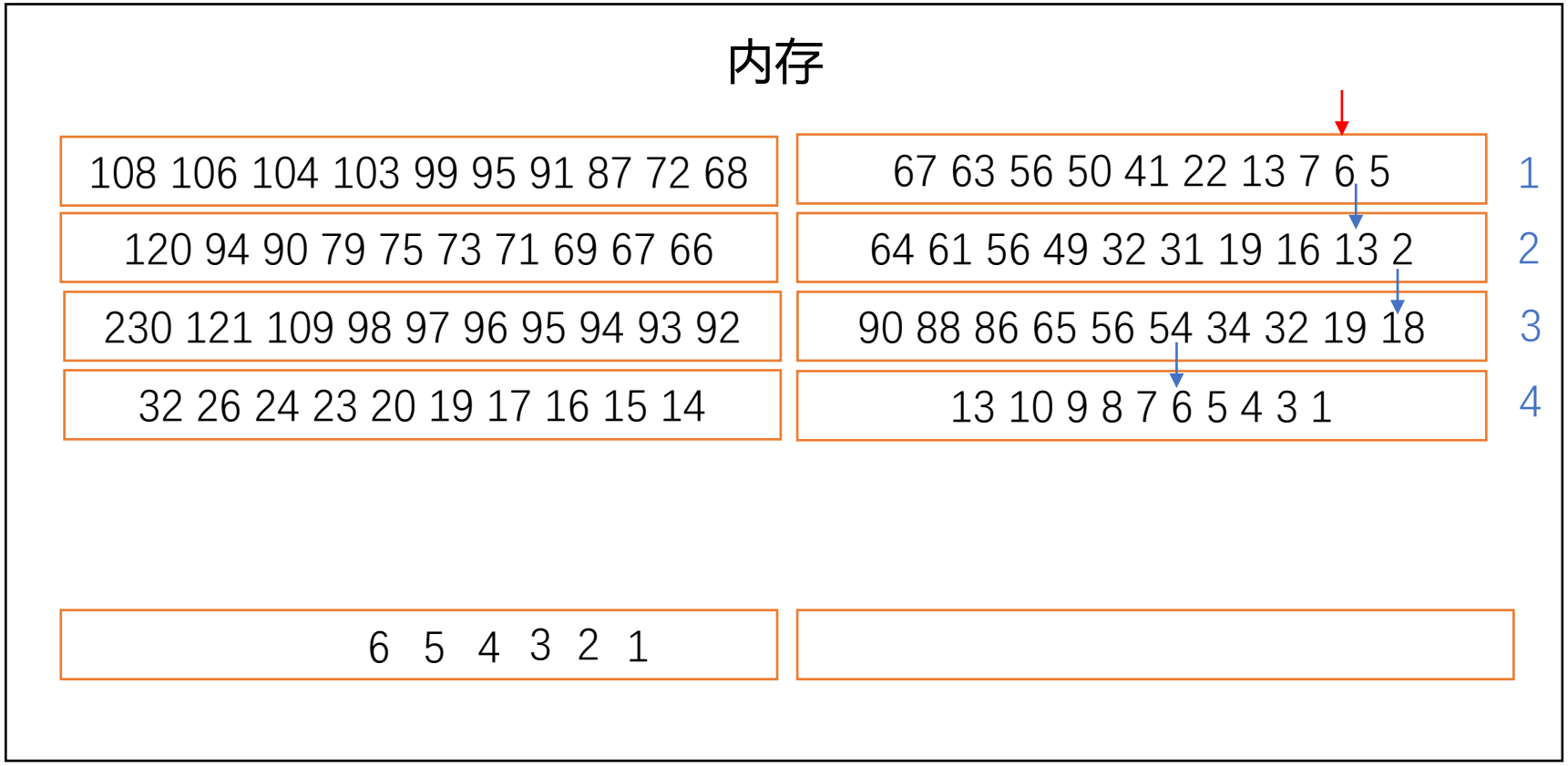


实验原理

基于排序的两趟扫描算法

第二趟：归并阶段

- 关系一元操作：去重复



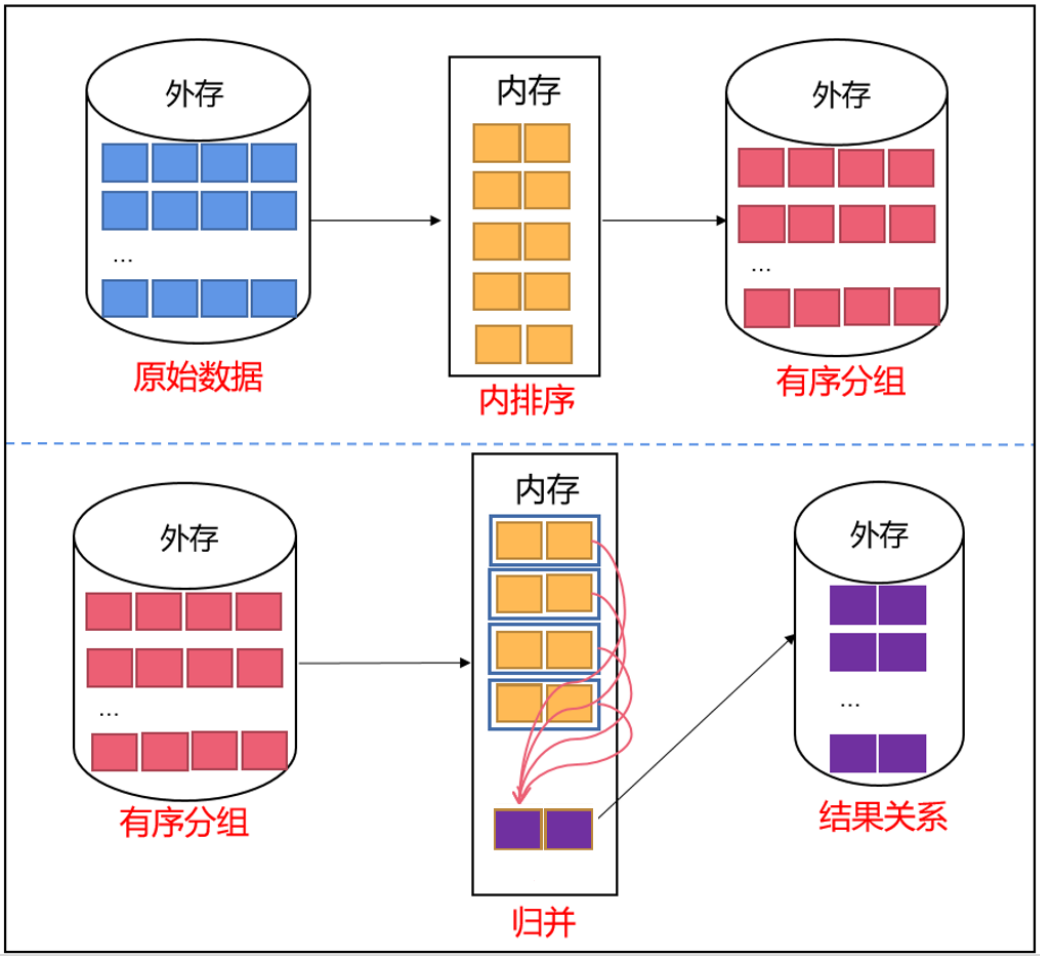
实验原理

基于排序的两趟扫描算法

第一趟：划分子表并进行子表排序

第二趟：归并阶段

- 关系一元操作
  - 去重复
  - 排序
  - 分组聚集
- 关系二元操作
  - 连接
  - 集合的并、交、差

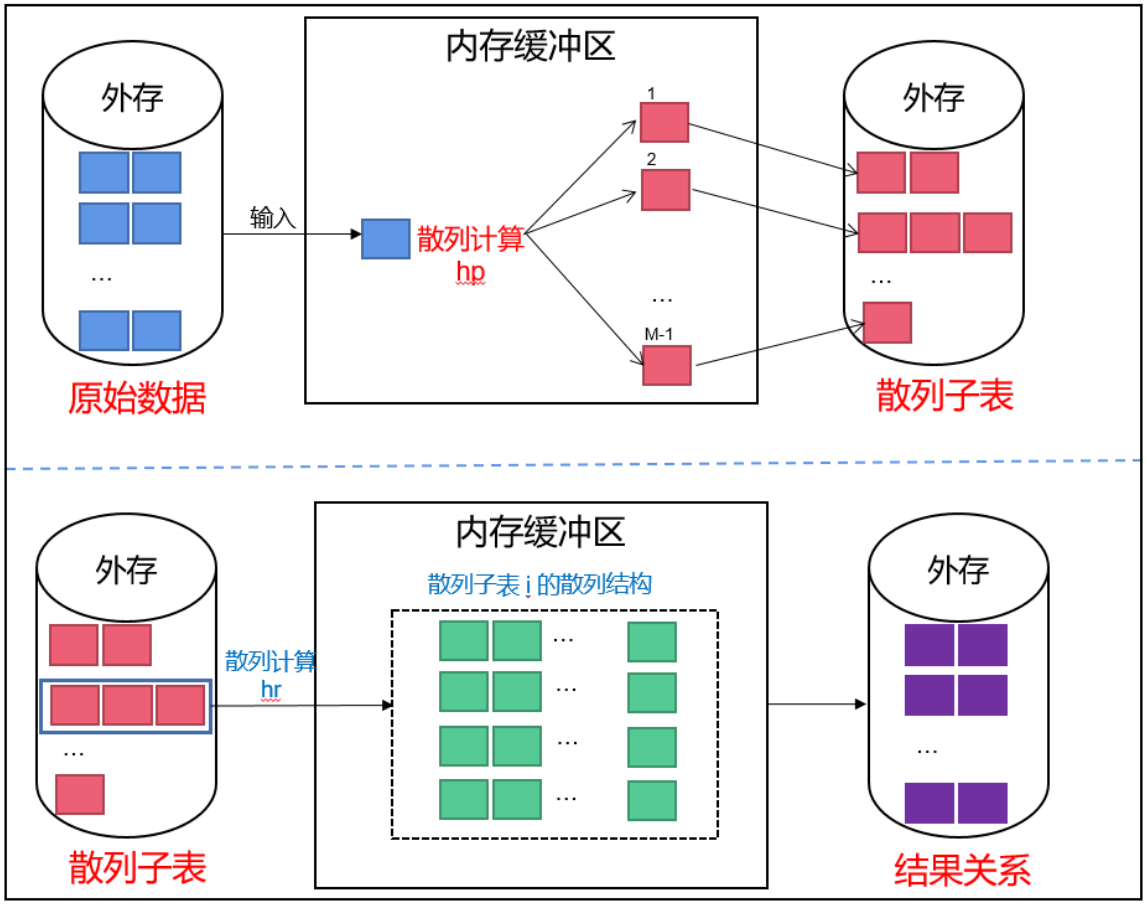


实验原理

基于散列的两趟扫描算法

第一趟：散列子表
用散列函数hp将原始关系划分到M-1个子表并存储，M为内存块数。

第二趟：处理每个子表
用另一个散列函数hr将子表读入内存，进行不同操作的处理。



实验数据

关系R具有两个属性A和B， A的值域为[100, 140]，B的值域为[400, 500]；

关系S具有两个属性C和D， C的值域为[120, 160]，D的值域为[420, 920]。

属性值均为int型（4个字节），R和S的每个元组的大小均为8个字节。

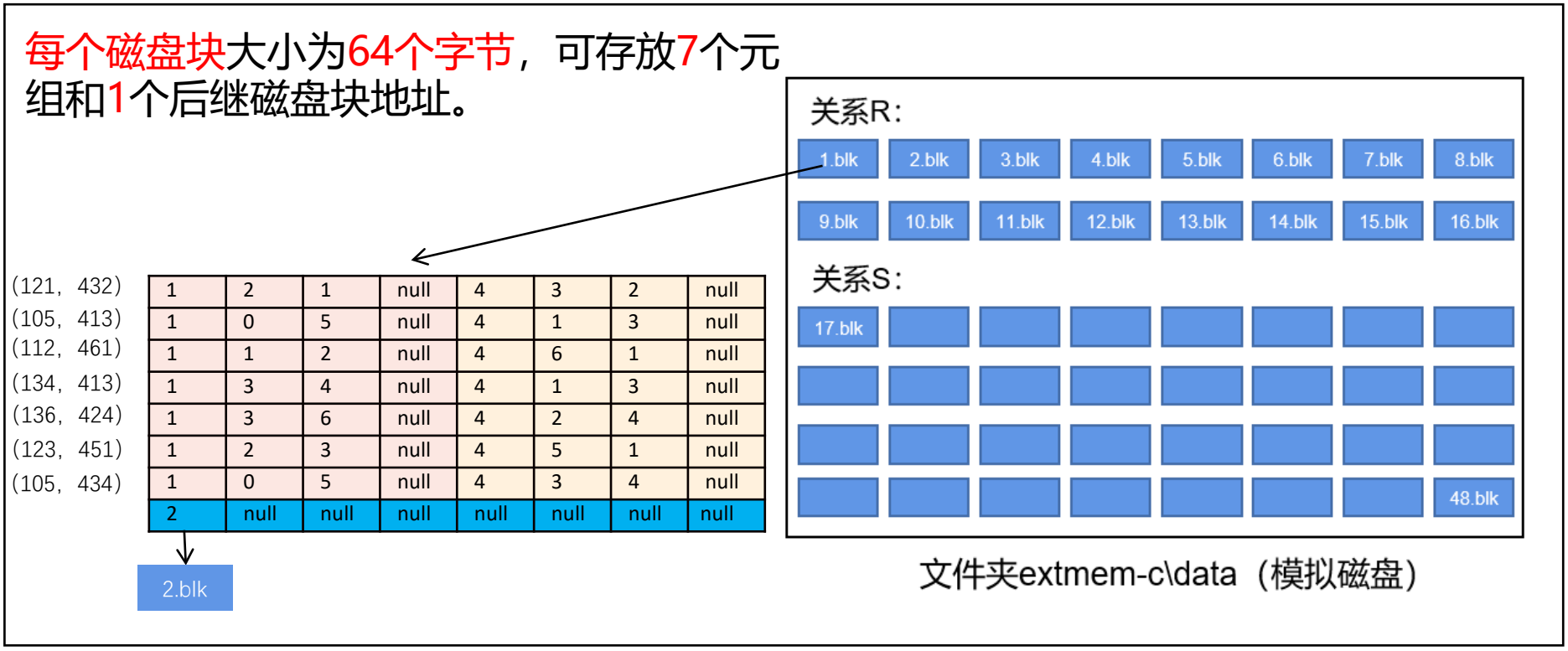
如： R的元组 **(125，463)，** 表示 R.A = 125； R.B = 463 。

1	2	5	null	4	6	3	null

实验数据

本实验已随机生成关系R和S，R中包含16 * 7 = 112个元组，S中包含32 * 7 = 224个元组。extmem-c\data下的每个文件模拟一个磁盘块。

每个磁盘块大小为64个字节，可存放7个元组和1个后继磁盘块地址。



ExtMem程序库

- ExtMem是C语言开发的模拟外存磁盘块存储和存取的程序库
- 功能包括：内存缓冲区管理、磁盘块读/写
- 提供了1个数据结构和7个API函数

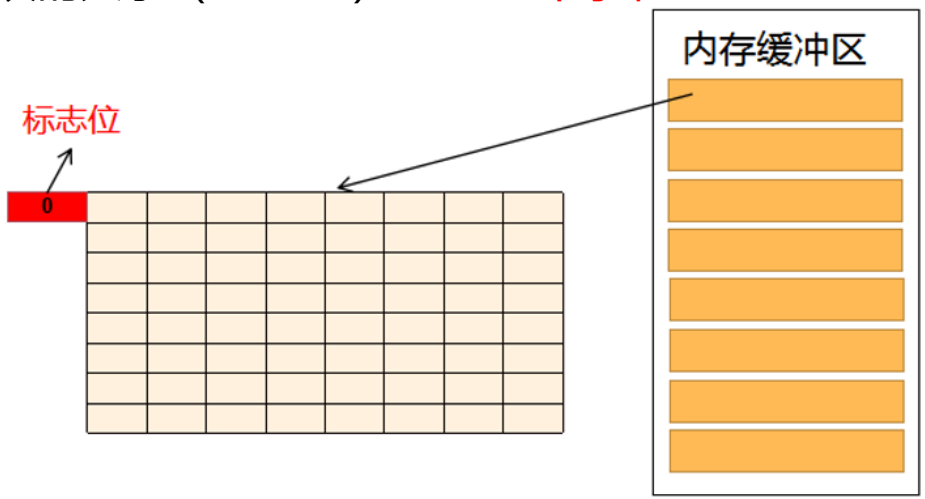
ExtMem程序库

1个数据结构:

Buffer数据类型:

numIO：外存I/O次数
bufSize：缓冲区大小（单位：字节）
blkSize：块的大小（单位：字节）
numAllBlk：缓冲区内可存放的最多块数
numFreeBlk：缓冲区内可用的块数
data：缓冲区内存区域(char *)

内存缓冲区
缓冲区大小（bufSize）：520个字节
块的大小（blkSize）：64个字节



缓冲区最多存放8个块：64 * 8 + 8 = 520，其中8个字节为标志位，表示每个块是否被占用。

文件test.c中给出了API函数使用的具体示例：

ExtMem程序库

7个API函数:

函数	功能
initBuffer	初始化缓冲区buf
getNewBlockInBuffer	在缓冲区中申请一个新的块
readBlockFromDisk	将磁盘上地址为addr的磁盘块读入缓冲区buf
writeBlockToDisk	将缓冲区buf内的块blk写入磁盘上地址为addr的磁盘块
freeBlockInBuffer	解除块对缓冲区内存的占用
dropBlockOnDisk	从磁盘上删除地址为addr的磁盘块内的数据
freeBuffer	释放缓冲区buf占用的内存空间

```c
int main(int argc, char **argv)
{
    Buffer buf; /* A buffer */
    unsigned char *blk; /* A pointer to a block */
    int i = 0;

    /* Initialize the buffer */
    if (!initBuffer(520, 64, &buf))
    {
        perror("Buffer Initialization Failed!\n");
        return -1;
    }

    /* Get a new block in the buffer */
    blk = getNewBlockInBuffer(&buf);

    /* Fill data into the block */
    for (i = 0; i < 8; i++)
        *(blk + i) = 'a' + i;

    /* Write the block to the hard disk */
    if (writeBlockToDisk(blk, 8888, &buf) != 0)
    {
        perror("Writing Block Failed!\n");
        return -1;
    }

    /* Read the block from the hard disk */
    if ((blk = readBlockFromDisk(1, &buf)) == NULL)
    {
        perror("Reading Block Failed!\n");
        return -1;
    }

    /* Process the data in the block */
    int X = -1;
    int Y = -1;
    int addr = -1;

    char str[5];
    printf("block 1:\n");
    for (i = 0; i < 7; i++) //一个blk存7个元组加一个地址
    {
        for (int k = 0; k < 4; k++)
        {
            str[k] = *(blk + i*8 + k);
        }
        X = atoi(str);
        for (int k = 0; k < 4; k++)
        {
            str[k] = *(blk + i*8 + 4 + k);
        }
        Y = atoi(str);
        printf("(%d, %d) ", X, Y);
    }
```

实验要求

基于ExtMem程序库，使用C语言模拟实现以下关系操作：

① 实现基于线性搜索的关系选择算法：基于ExtMem程序库，使用C语言实现线性搜索算法，选出**S.C=128**的元组，记录IO读写次数，并将选择结果存放在磁盘上。（模拟实现 select S.C, S.D from S where S.C = 128）

② 实现两阶段多路归并排序算法（TPMMS）：利用内存缓冲区将关系R和S分别排序，并将排序后的结果存放在磁盘上。

③ 实现基于索引的关系选择算法：利用（2）中的排序结果为关系R或S分别建立索引文件，利用索引文件选出**S.C=128**的元组，并将选择结果存放在磁盘上。记录IO读写次数，与（1）中的结果对比。（模拟实现 select S.C, S.D from S where S.C = 128 ）

④ 实现基于排序的连接操作算法（Sort-Merge-Join）：对关系S和R计算S.C连接R.A，并统计连接次数，将连接结果存放在磁盘上。（模拟实现 select S.C, S.D, R.A, R.B from S inner join R on S.C = R.A）

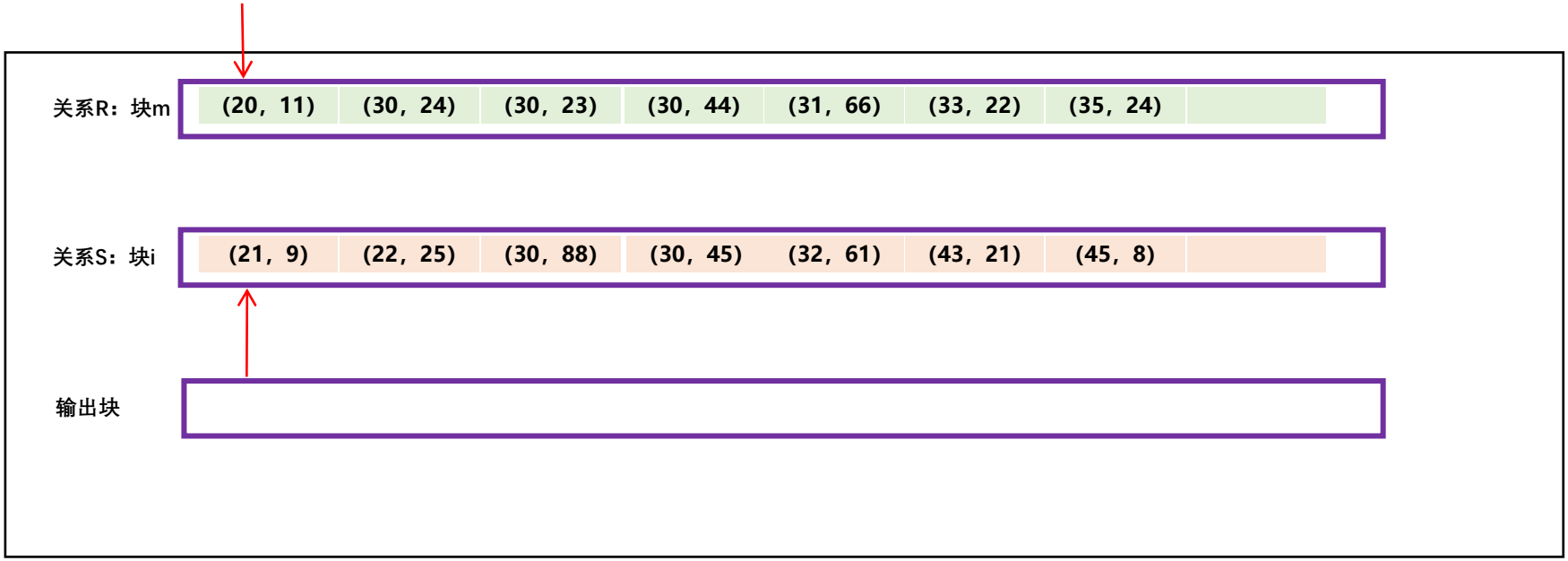
⑤ 实现基于排序或散列的两趟扫描算法，实现其中一种集合操作算法：并（S ∪ R）、交（S ∩ R）、差（S - R）中的一种。将结果存放在磁盘上，并统计并、交、差操作后的元组个数。

实验要求

* **（附加题）基于排序或散列的两趟扫描算法，实现剩余的两种集合操作算法： 并、交、差。**
* 将结果存放在磁盘上，并统计并、交、差操作后的剩余元组个数。

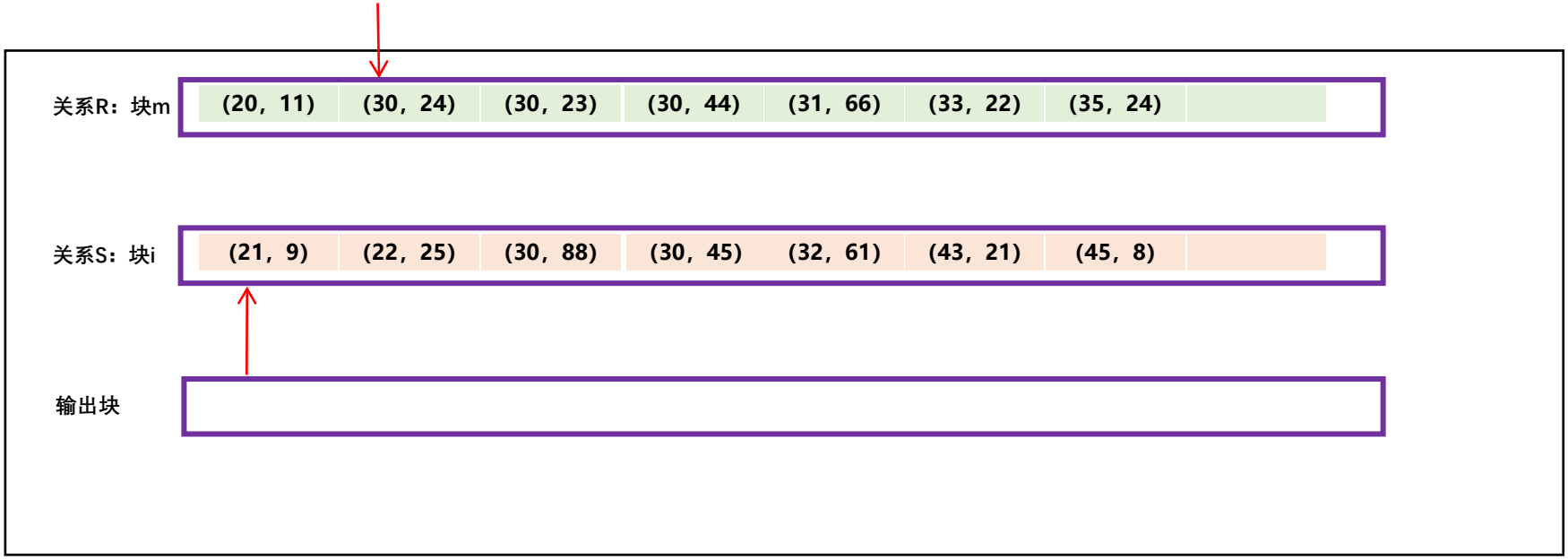
实验要求

实现基于排序的连接操作算法（Sort-Merge-Join）：
模拟实现 select S.C, S.D, R.A, R.B from S inner join R on S.C = R.A



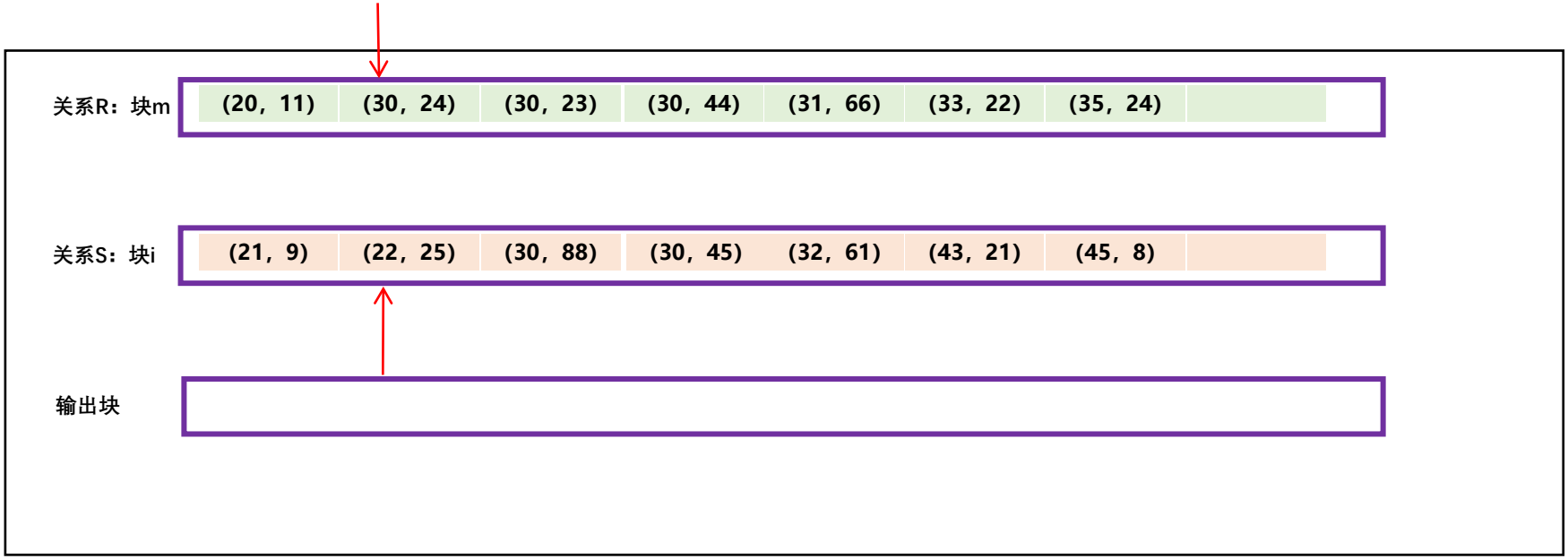
实验要求

实现基于排序的连接操作算法（Sort-Merge-Join）：
模拟实现 select S.C, S.D, R.A, R.B from S inner join R on S.C = R.A



实验要求

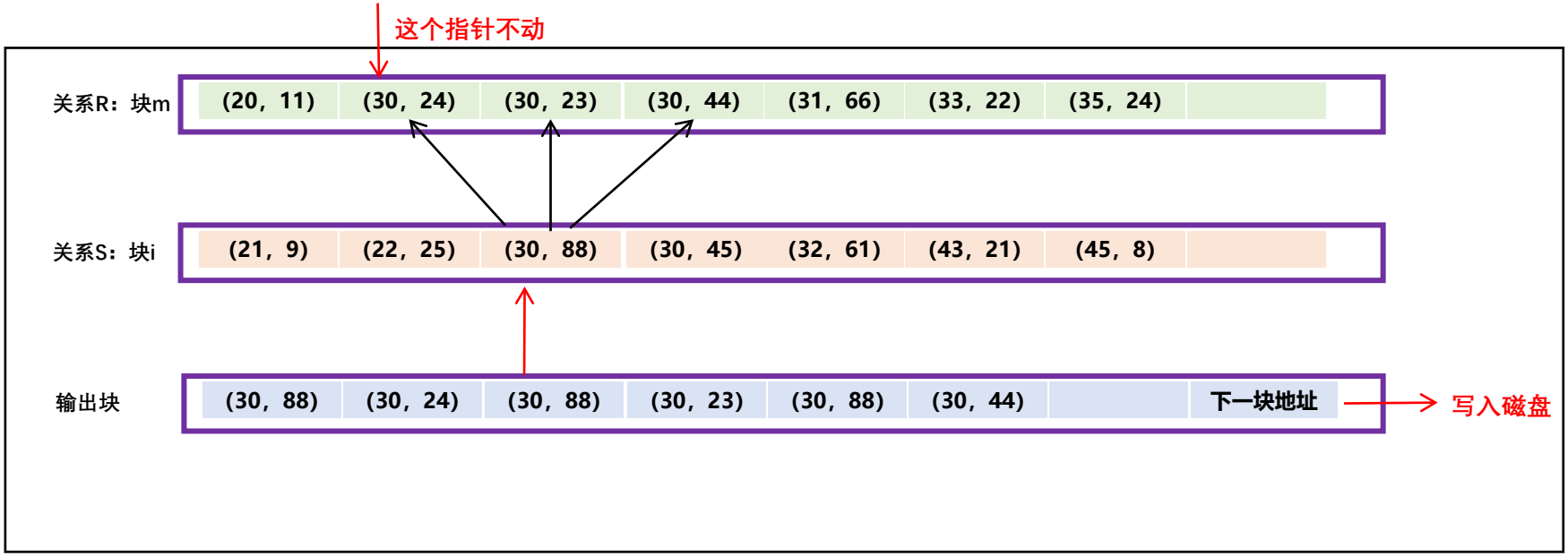
实现基于排序的连接操作算法（Sort-Merge-Join）：
模拟实现 select S.C, S.D, R.A, R.B from S inner join R on S.C = R.A



实验要求

实现基于排序的连接操作算法（Sort-Merge-Join）：

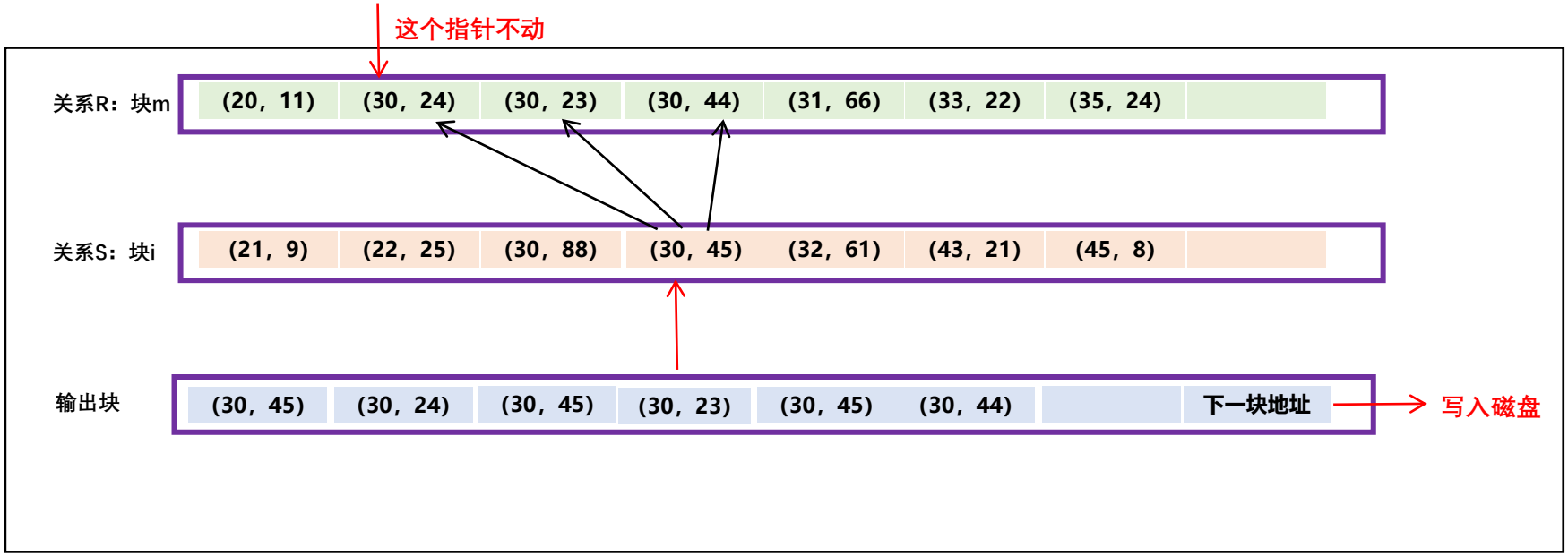
模拟实现 select S.C, S.D, R.A, R.B from S inner join R on S.C = R.A



实验要求

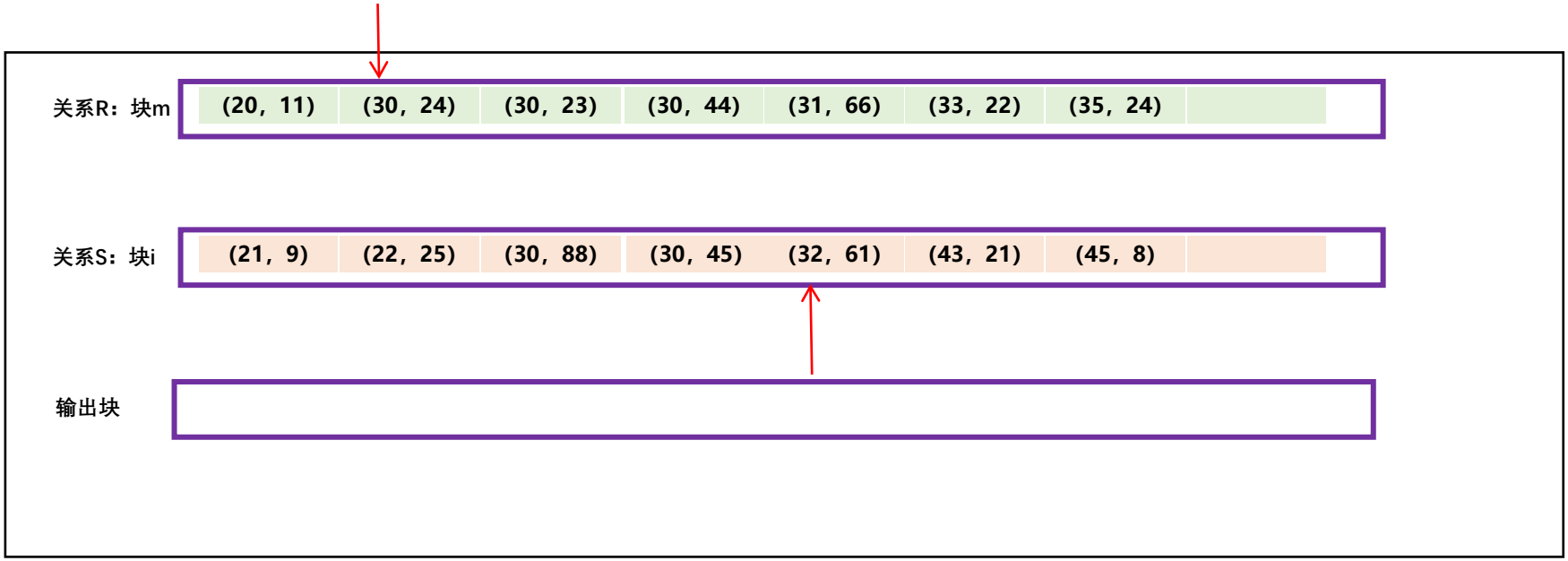
实现基于排序的连接操作算法（Sort-Merge-Join）：

模拟实现 select S.C, S.D, R.A, R.B from S inner join R on S.C = R.A



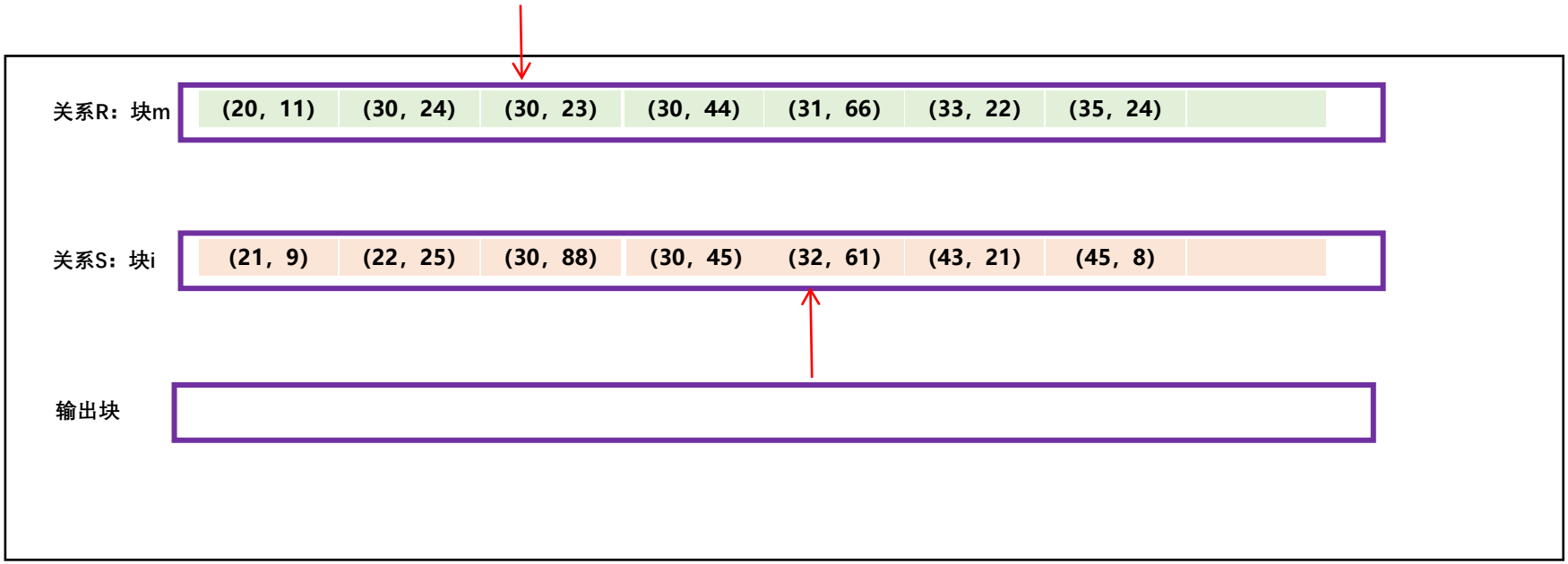
实验要求

实现基于排序的连接操作算法（Sort-Merge-Join）：
模拟实现 select S.C, S.D, R.A, R.B from S inner join R on S.C = R.A



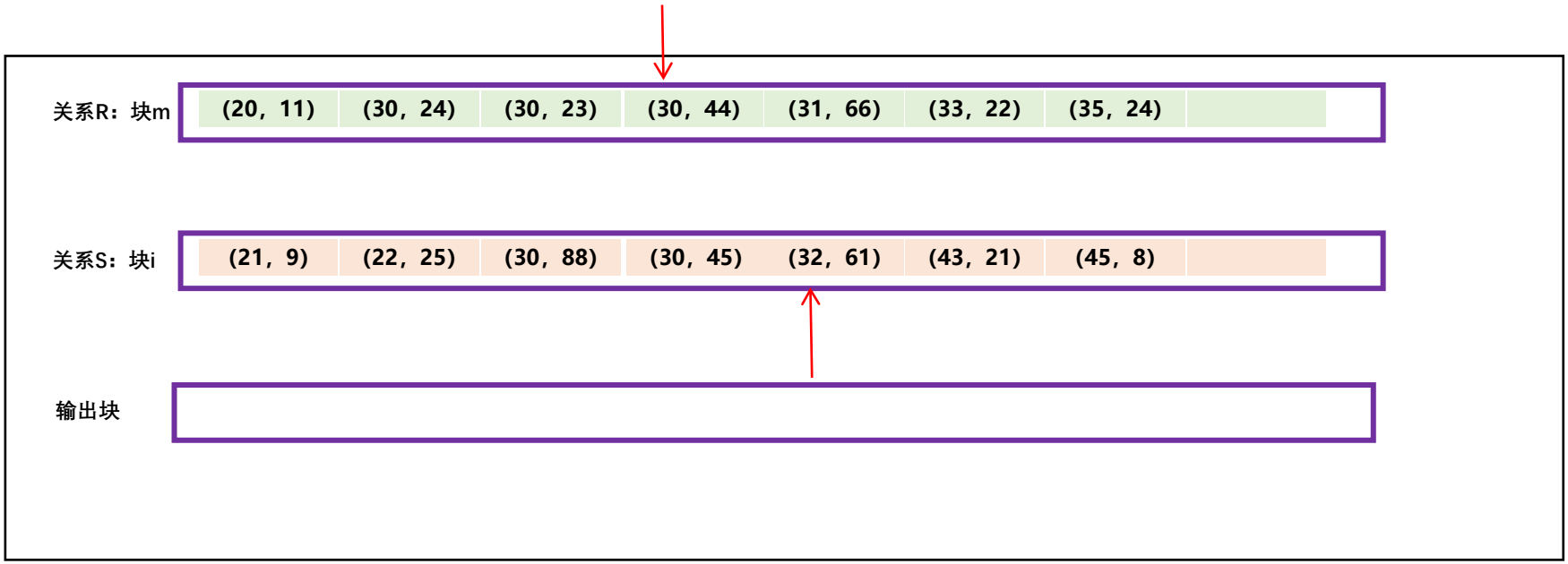
实验要求

实现基于排序的连接操作算法（Sort-Merge-Join）：
模拟实现 select S.C, S.D, R.A, R.B from S inner join R on S.C = R.A



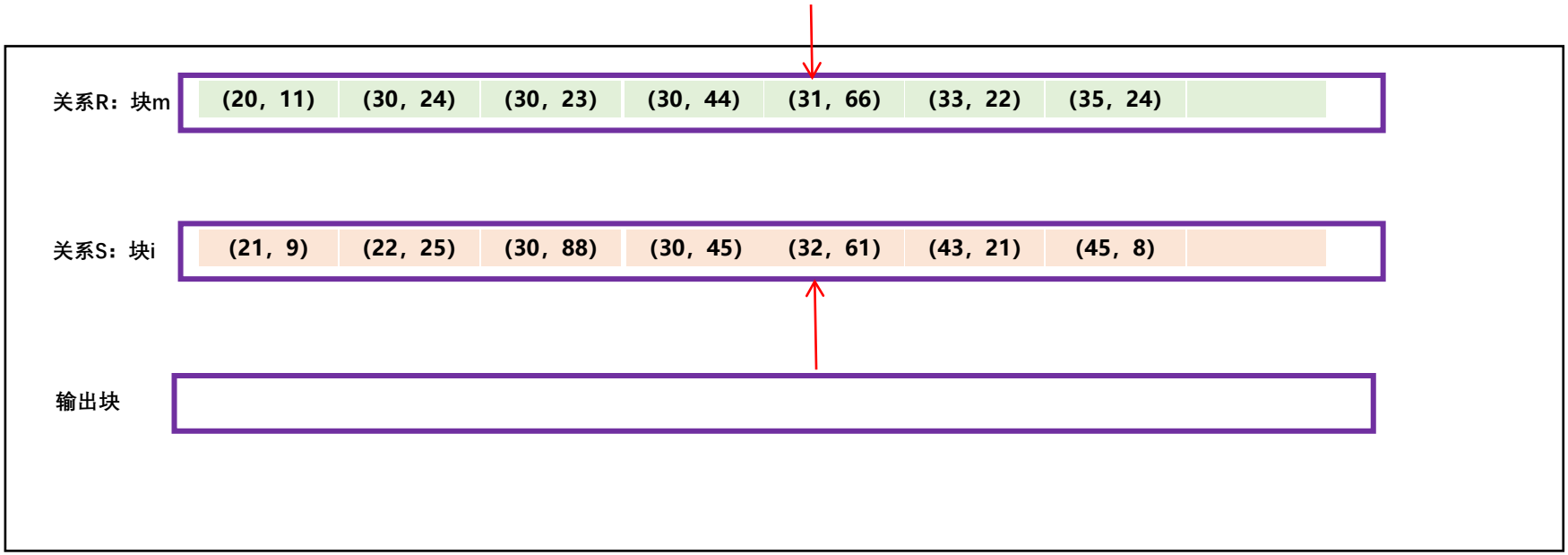
实验要求

实现基于排序的连接操作算法（Sort-Merge-Join）：
模拟实现 select S.C, S.D, R.A, R.B from S inner join R on S.C = R.A



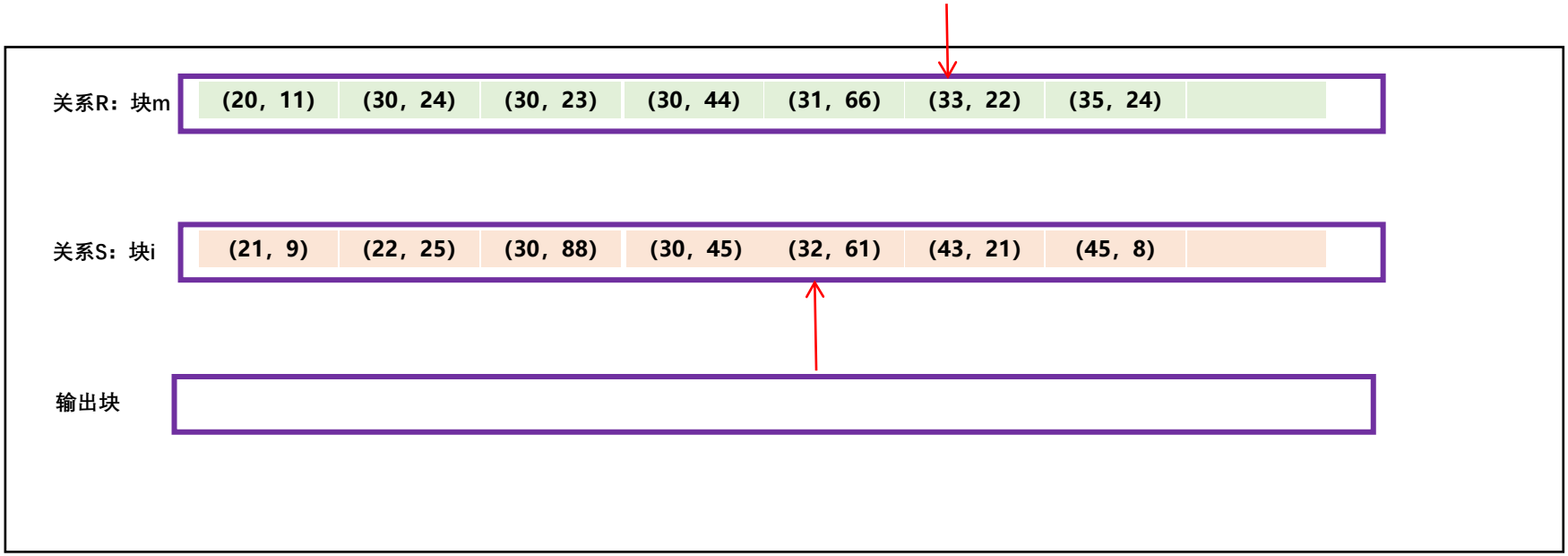
实验要求

实现基于排序的连接操作算法（Sort-Merge-Join）：
模拟实现 select S.C, S.D, R.A, R.B from S inner join R on S.C = R.A



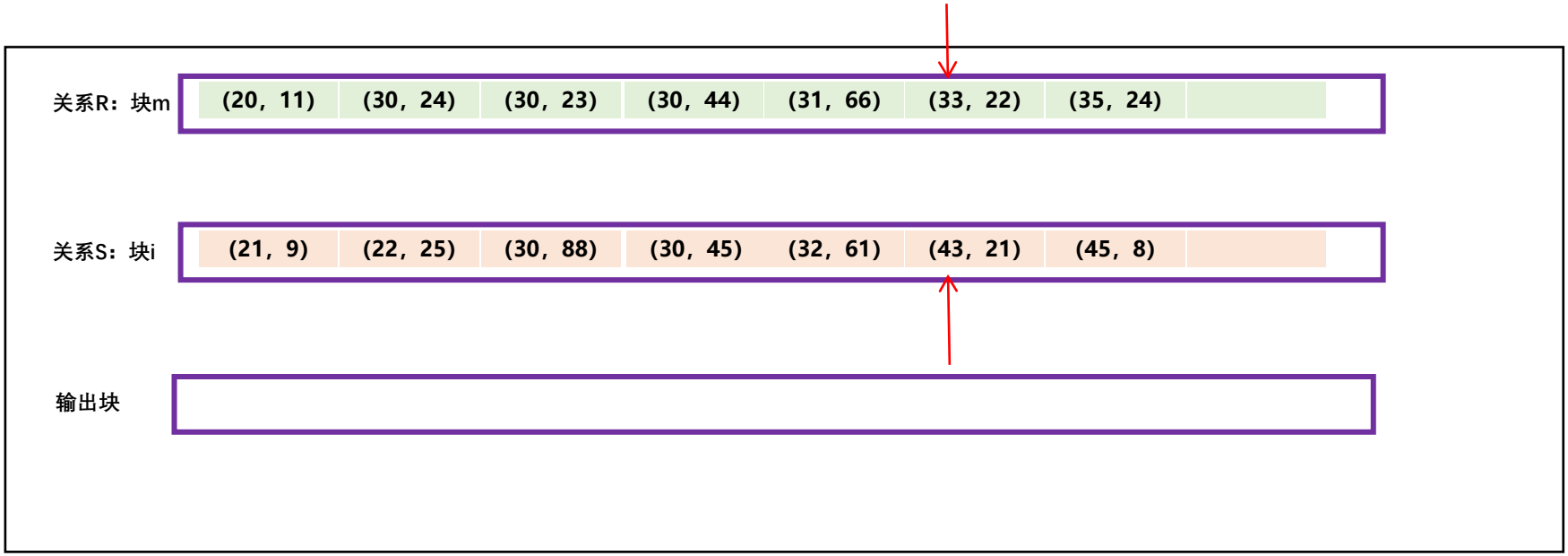
实验要求

实现基于排序的连接操作算法（Sort-Merge-Join）：
模拟实现 select S.C, S.D, R.A, R.B from S inner join R on S.C = R.A



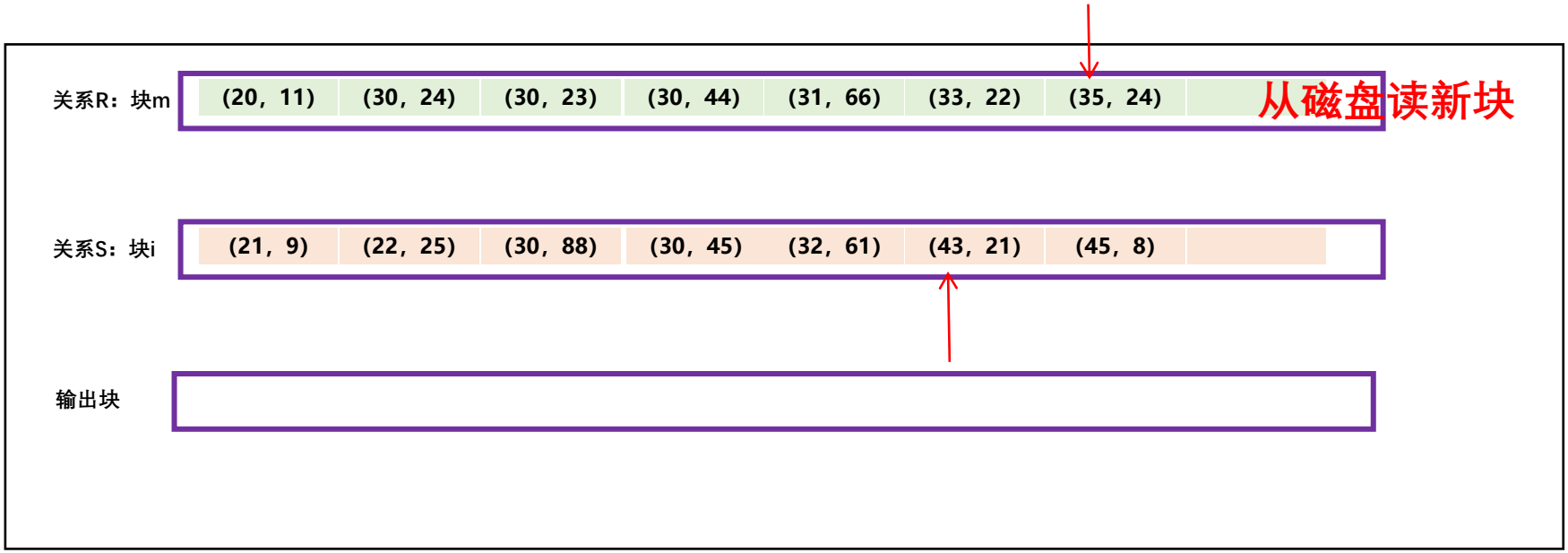
实验要求

实现基于排序的连接操作算法（Sort-Merge-Join）：
模拟实现 select S.C, S.D, R.A, R.B from S inner join R on S.C = R.A



实验要求

实现基于排序的连接操作算法（Sort-Merge-Join）：
模拟实现 select S.C, S.D, R.A, R.B from S inner join R on S.C = R.A



实验要求

输出范例

```
----------------------------------------
基于线性搜索的选择算法 R.A=30:
----------------------------------------
读入数据块1
读入数据块2
读入数据块3
(X=30, Y=913)
读入数据块4
读入数据块5
读入数据块6
(X=30, Y=624)
读入数据块7
读入数据块8
读入数据块9
读入数据块10
读入数据块11
读入数据块12
读入数据块13
读入数据块14
读入数据块15
(X=30, Y=617)
读入数据块16
(X=30, Y=703)
注：结果写入磁盘：100

满足选择条件的元组一共4个。

IO读写一共 17次。
```

```
----------------------------------------
基于索引的选择算法 R.A=30:
----------------------------------------
读入索引块350
没有满足条件的元组。
读入索引块351
读入数据块312
(X=30, Y=913)
(X=30, Y=624)
(X=30, Y=617)
读入数据块313
(X=30, Y=703)
注：结果写入磁盘：120

满足选择条件的元组一共4个。

IO读写一共 5次。
```

对比IO读写次数，理解索引的作用。

仅参考输出样式，
实验数据已更新！

实验要求

输出范例

```
------------------------------------
基于排序的连接算法：
------------------------------------
注：结果写入磁盘：701
注：结果写入磁盘：702
...
注：结果写入磁盘：760
注：结果写入磁盘：761
注：结果写入磁盘：762
注：结果写入磁盘：763

总共连接220次。
```

```
------------------------------------
基于排序的集合的交运算：
------------------------------------
(X=36,Y=895)
(X=22,Y=712)
(X=30,Y=624)
(X=23,Y=758)
(X=30,Y=703)
(X=30,Y=617)
(X=25,Y=440)
注：结果写入磁盘：140
(X=40,Y=557)
(X=34,Y=665)
注：结果写入磁盘：141

S和R的交集有9个元组。
```

```
------------------------------------
基于排序的集合的并运算：
------------------------------------
注：结果写入磁盘：801
注：结果写入磁盘：802
注：结果写入磁盘：803
...
注：结果写入磁盘：845
注：结果写入磁盘：846
注：结果写入磁盘：847

R和S的并集有327个元组。
```

```
------------------------------------
基于排序的集合的差运算：
------------------------------------
注：结果写入磁盘：901
注：结果写入磁盘：902
注：结果写入磁盘：903
...
注：结果写入磁盘：928
注：结果写入磁盘：929
注：结果写入磁盘：930
注：结果写入磁盘：931

S和R的差集(S-R)有215个元组。
```

仅参考输出样式，
实验数据已更新！

各任务分值

基本任务（100 分）

①基于线性搜索的关系选择 (10分)
②两阶段多路归并排序算法 (40分)
③基于索引的关系选择算法 (10分)
④基于排序的连接操作算法 (20分)
⑤基于排序或散列的两趟扫描算法 (20分)

附加题（10 分）

集合操作算法：并、交、差中剩余的两种 。 (每种算法5分，选做)

提交方式

课后提交：

- 将实验报告、工程文件打成zip 包，提交至作业提交平台（截止日期参考平台发布）

  作业平台入口：http://grader.tery.top:8000/#/login

- **统一用codeblocks**

同学们
请开始实验吧！