

# 一类搜索的优化思想

## ——数据有序化

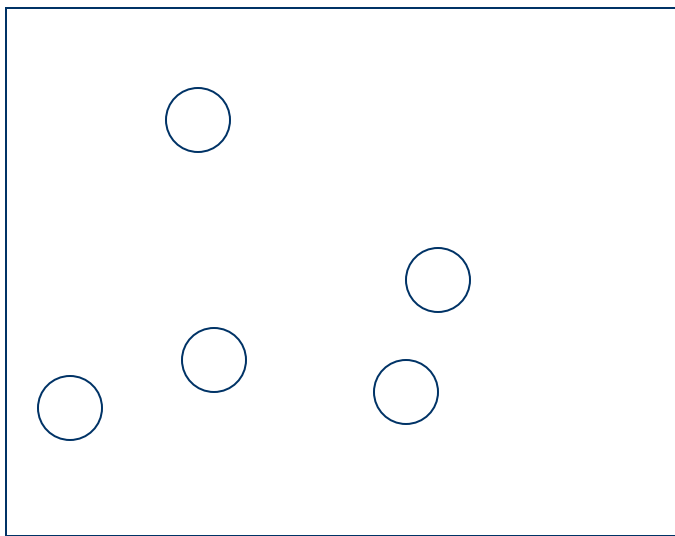
南京市金陵中学 刘一鸣

# 数据有序化

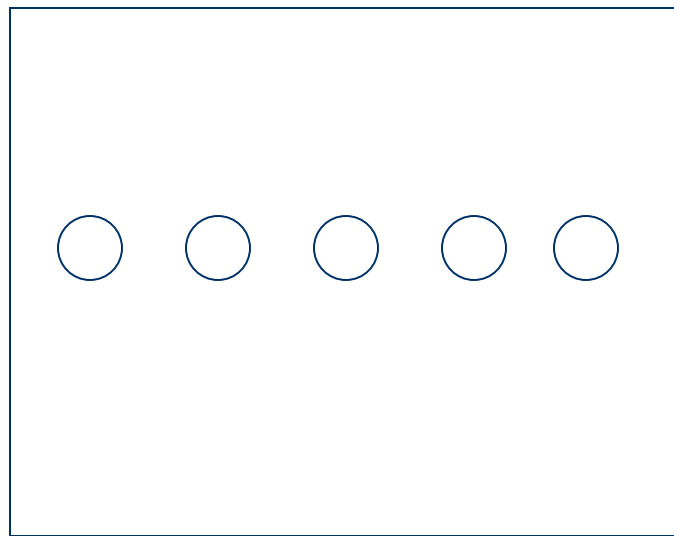
数据有序化的思想，就是将杂乱的数据，通过简单的分类和排序，变成有序的数据，从而加快搜索的速度。

- ◆ 为什么要进行数据有序化
- ◆ 数据有序化的实现
- ◆ 两种实现方法的比较

# 为什么要进行数据有序化



杂乱的数据



有序的数据

# 例1 装箱问题

题目大意：

现有一个体积为 $V$ 的集装箱和 $N$ 种货物，每一种货物都有固定的体积，数量无限。你的任务是：写一个程序，求出最少用多少个货物，就能放满集装箱。

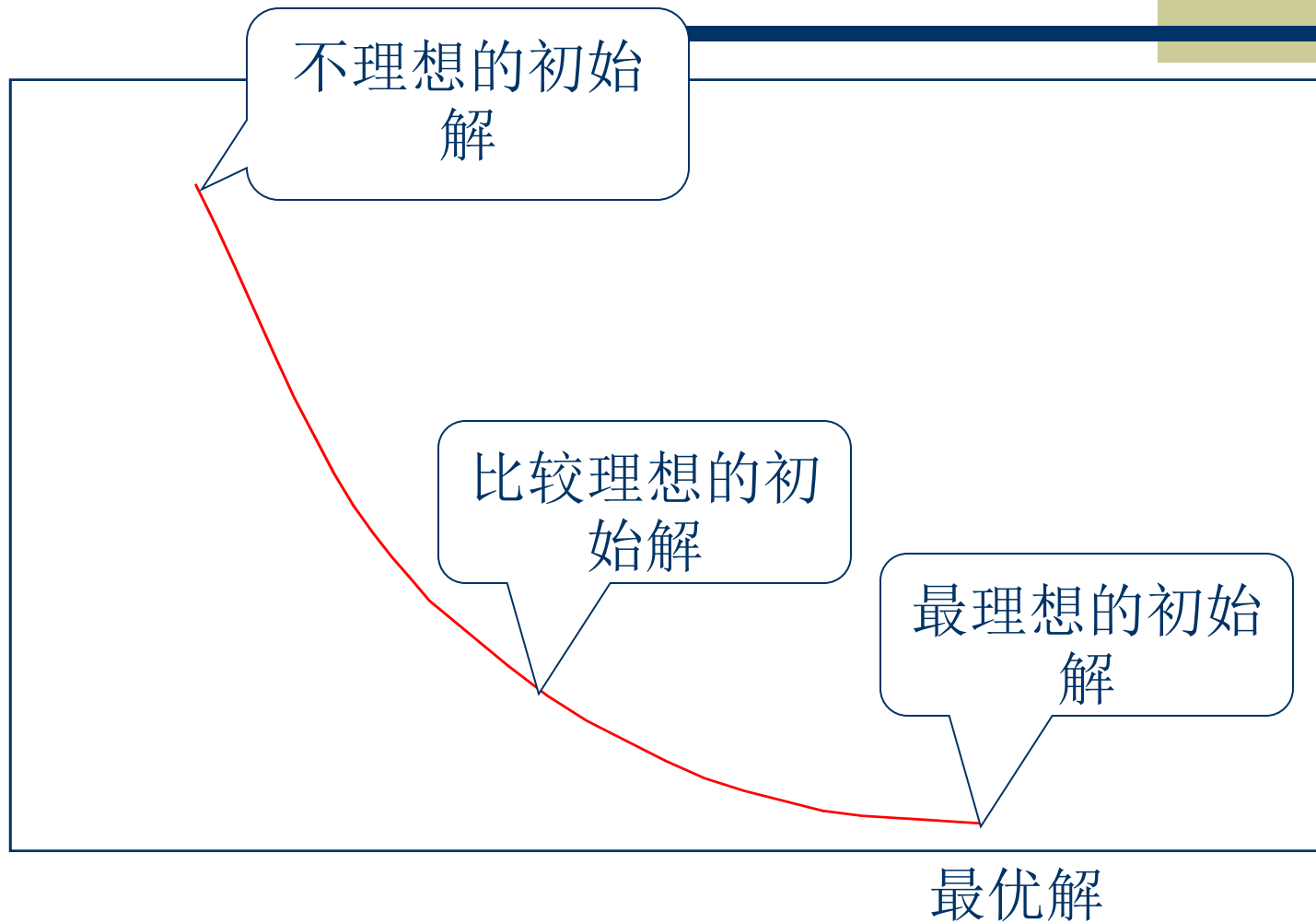
数据规模：  $V_{\text{货}} \leq V \leq 10^9$

# 运行时间的对比

测试方法：随机生成20个数据，测试运行时间并求平均值。

N	不排序，直接搜索	先按体积从大到小排序，再搜索
10	>160秒	9.8545秒
30	>200秒	0.1356秒
60	>200秒	0.1595秒
100	>200秒	0.2285秒

# 程序效率不同的原因



# 数据有序化的益处

- 对于大多数的数据，都有良好的优化效果；
- 简便易行；
- 和其他类型的优化方法一般都不冲突。



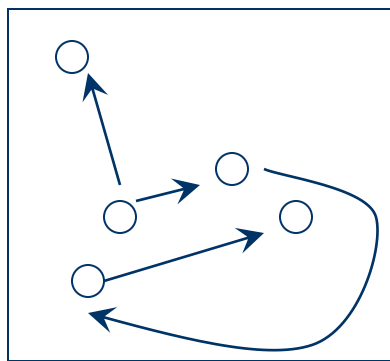
# 数据有序化的实现

- ◆ 预处理阶段的数据有序化
- ◆ 实时处理阶段的数据有序化





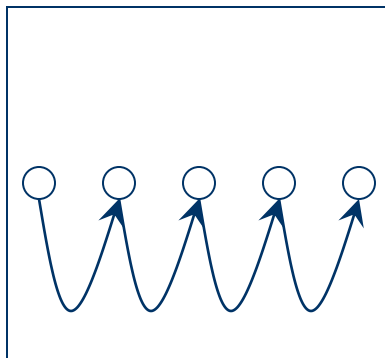
# 预处理阶段的数据有序化



杂乱的数据

常规方法

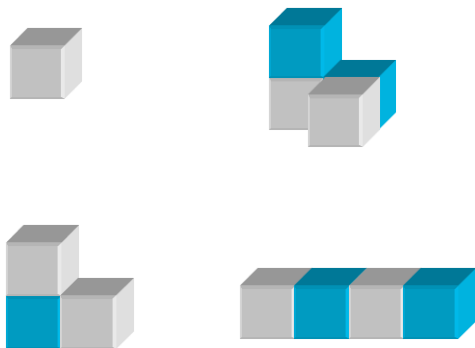
加工



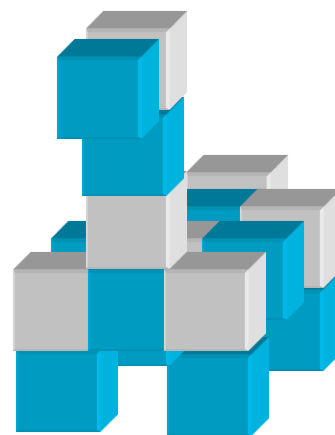
有序的数据

## 例2 积木搭建

- ◆ 题目大意：给定12种积木和一个体积小于50的构型，求最少使用多少个积木可以将这个构型搭建起来

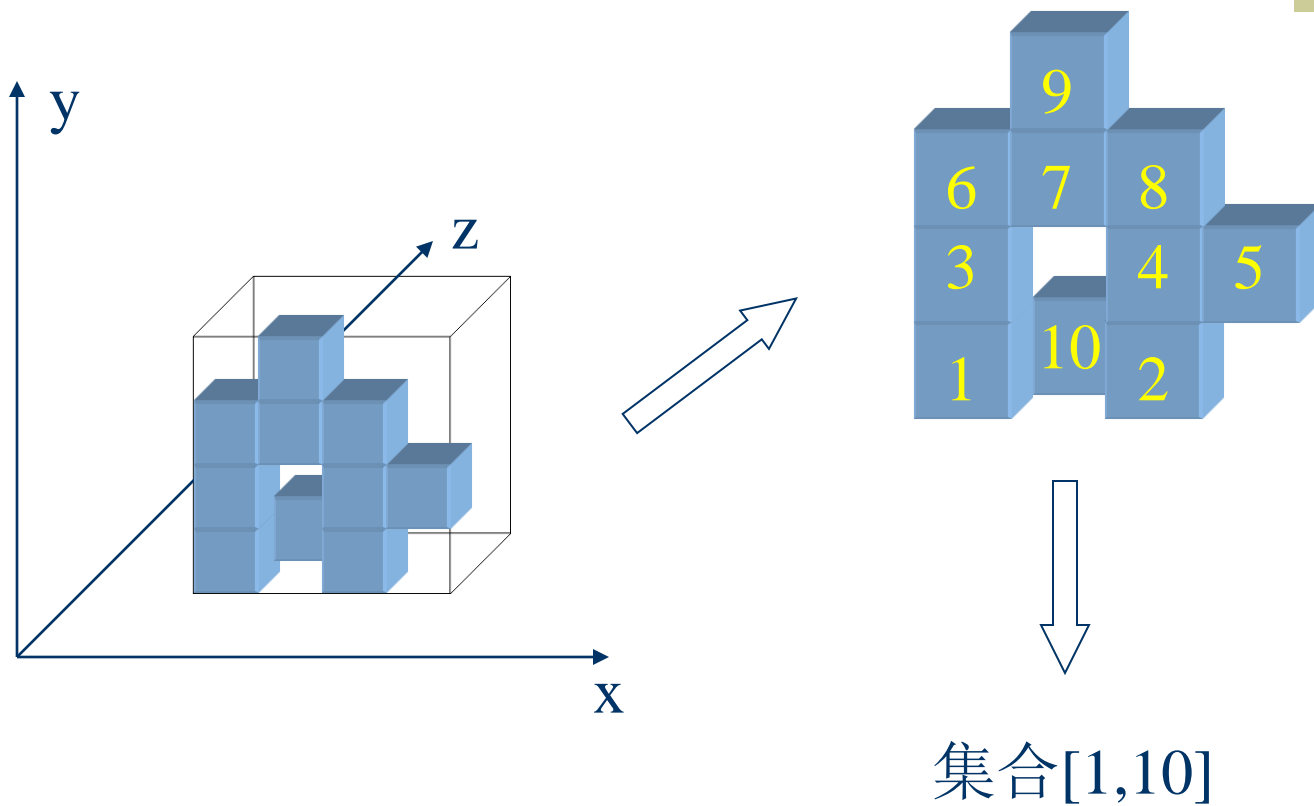


积木示例

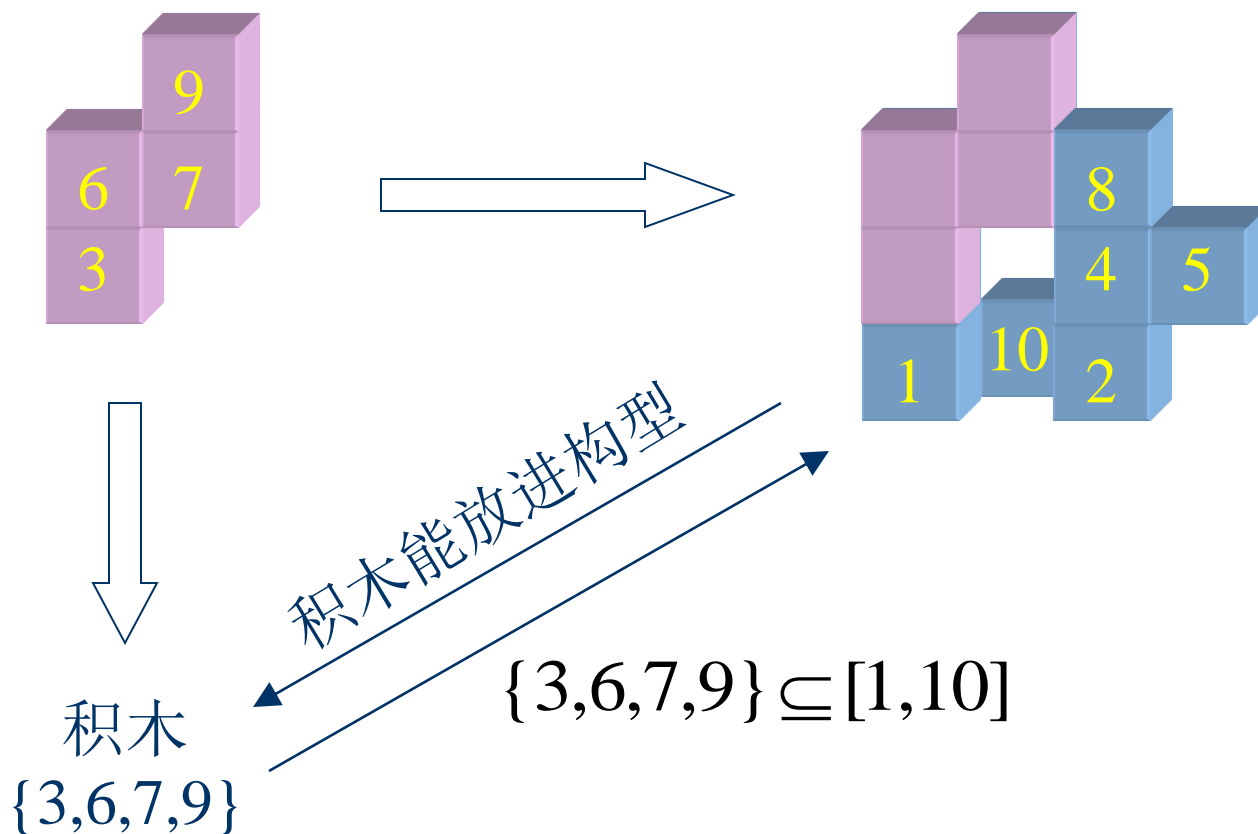


目标构型示例

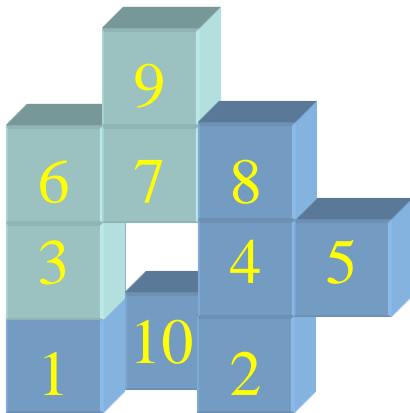
# 第1步数据有序化



## 第2步数据有序化



# 从构型中挖去一个积木



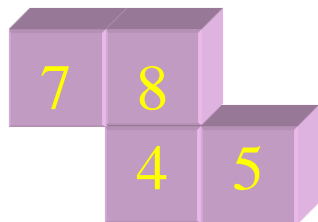
$[1,10]$



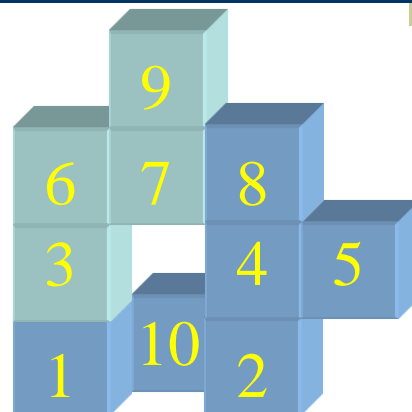
$\{1,2,4,5,8,10\}$

$$[1,10] - \{3,6,7,9\} = \{1,2,4,5,8,10\}$$

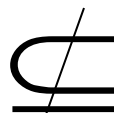
## 试图再放一块积木



$\{4,5,7,8\}$

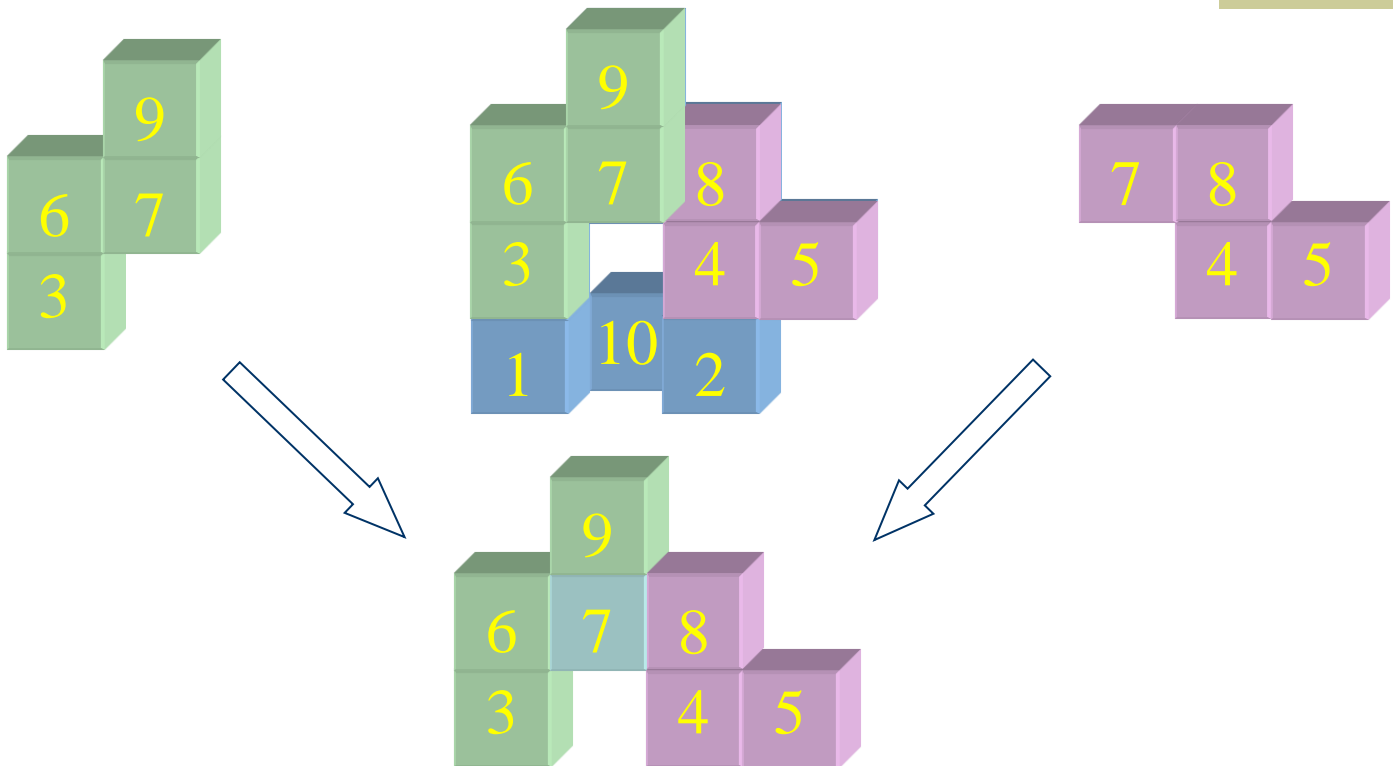


$\{1,2,4,5,8,10\}$



积木不能放入构型

# 积木的冲突



$$\{3, 6, 7, 9\} \cap \{4, 5, 7, 8\} = \{7\} \neq \varnothing$$

# 数据有序化前后 数学模型的对比

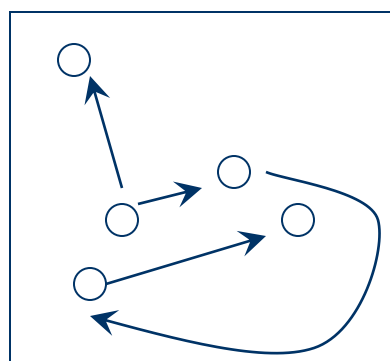
数据有序化前	数据有序化后
目标构型（3维）	目标集合（1维）
积木（3维）	小集合（1维）
积木拼接成为目标构型	小集合的合并成为目标集合
积木在3维空间里没有冲突	小集合的交集为空集

数据有序化后，数学模型得到了精简





# 实时处理阶段的数据有序化



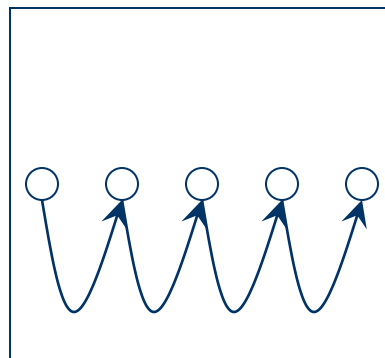
杂乱的数据

常规方法

加工

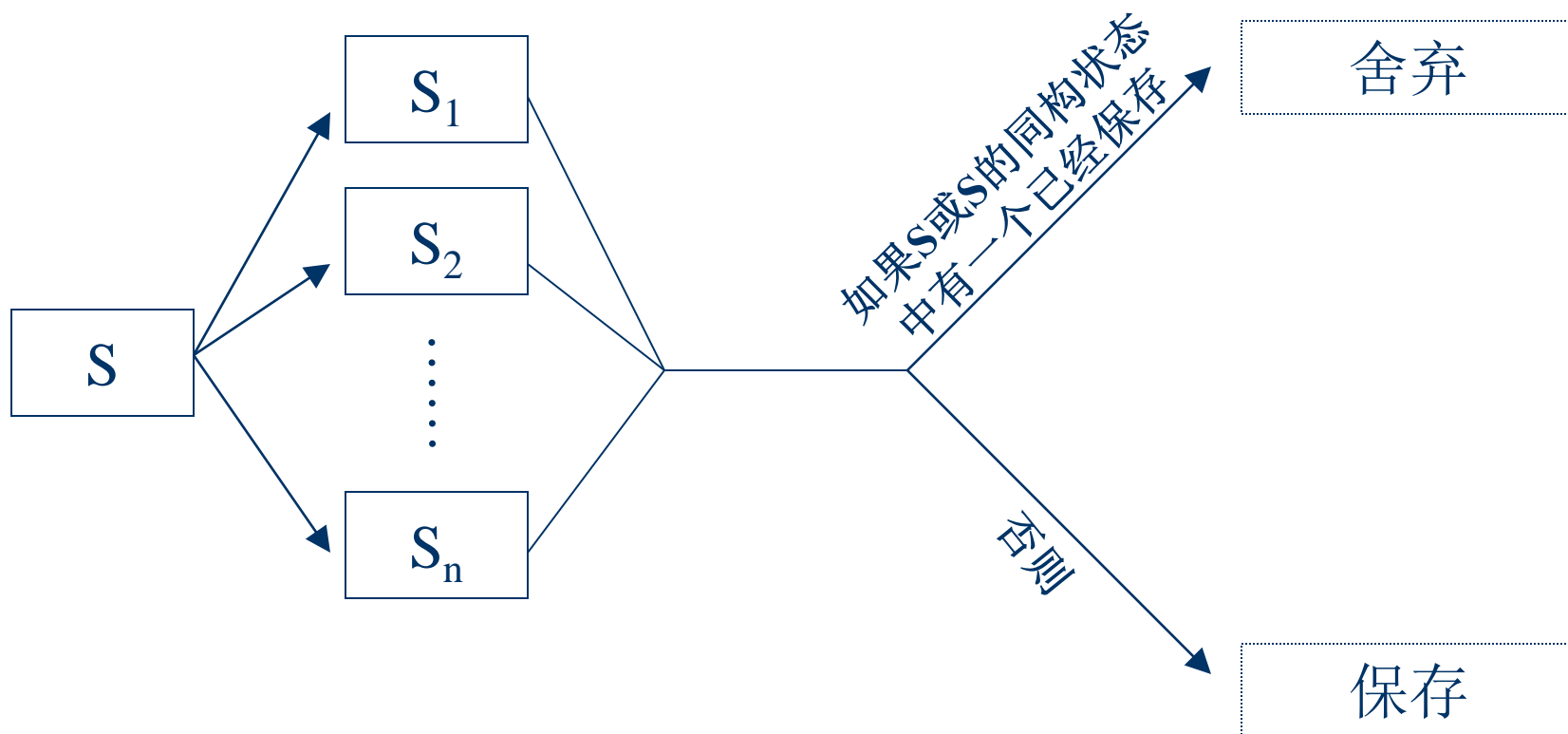
保存

舍弃

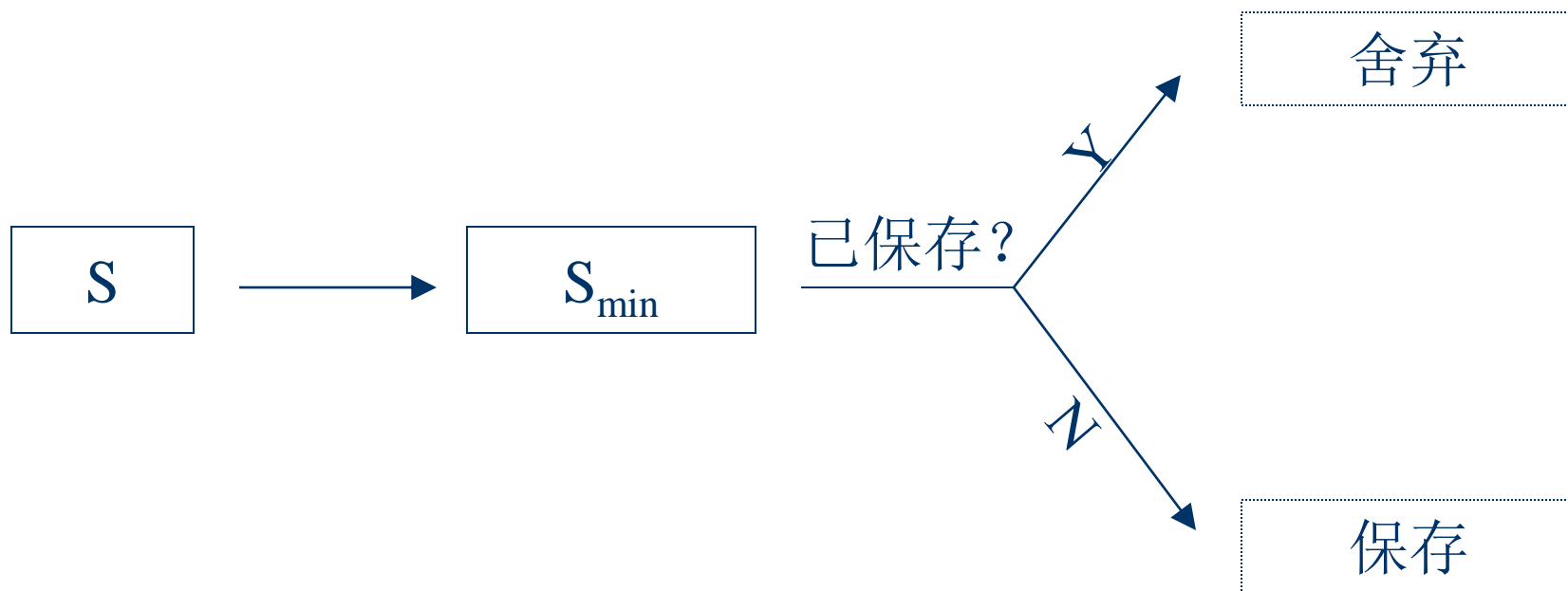


有序的数据

# 传统表示方法



# 最小表示法



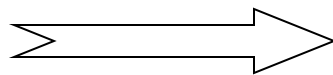
## 例3 N皇后问题-2

- ◆ 题目大意：假定通过翻转、旋转得到的状态与原状态属于同构状态，求所有不同构的 $n$ 皇后状态总数。

## 状态表示方法

- ◆ 由于一行中只能有一个皇后，所以用一个  $n$  元组  $(a_1, a_2, a_3, \dots, a_n)$  表示当前的状态，其中  $a_i$  表示第  $i$  列的皇后所在的行。

	Q			
			Q	
Q				
		Q		
				Q



$(3, 1, 4, 2, 5)$

# 翻转、旋转的具体过程(1)

以铅垂线为轴的翻转：

	$Q_1$			
			$Q_2$	
$Q_3$				
		$Q_4$		
				$Q_5$

			$Q_1$	
	$Q_2$			
				$Q_3$
		$Q_4$		
$Q_5$				

$(a_1, a_2, a_3, a_4, a_5)$

→

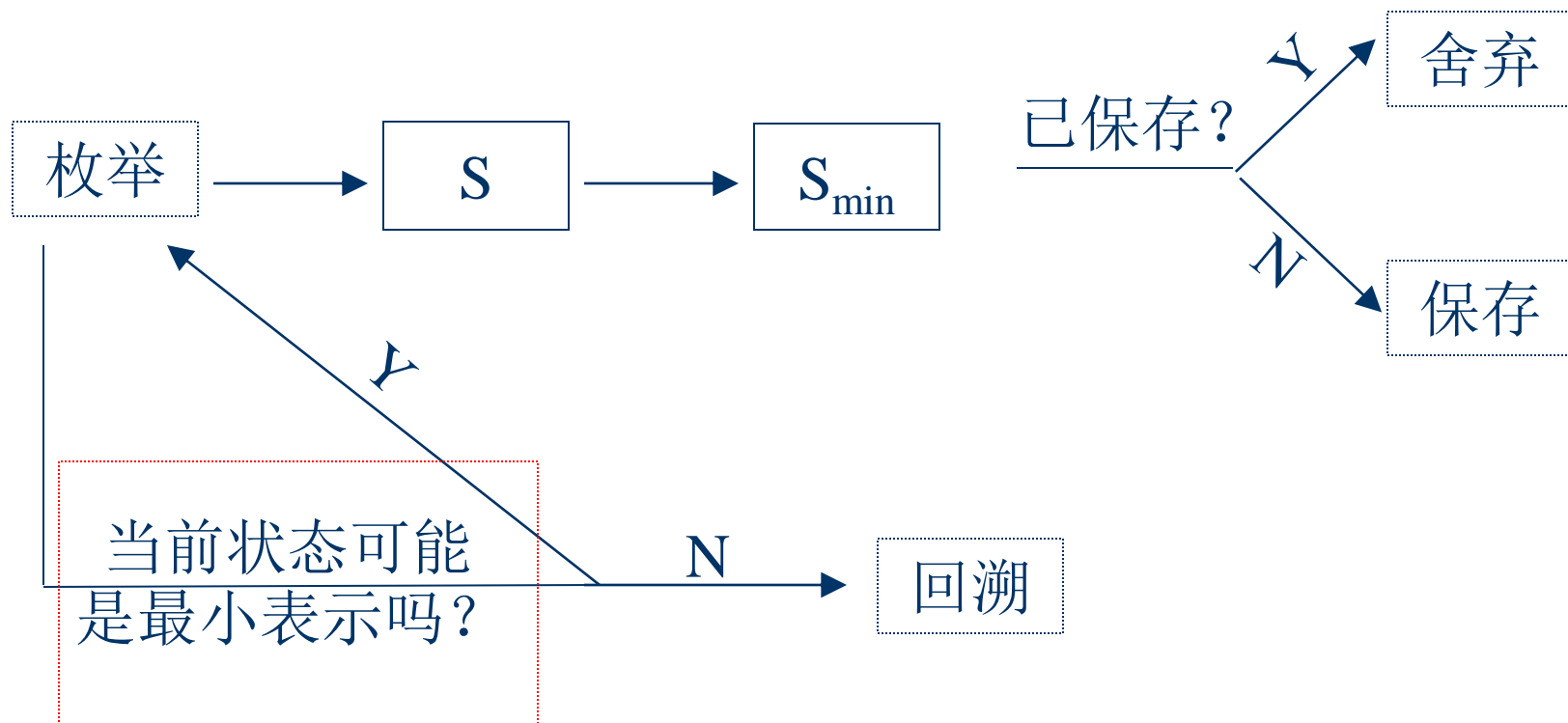
$(a_5, a_4, a_3, a_2, a_1)$

# 翻转、旋转的具体过程(2)

以原角线为轴的翻转

$$\begin{aligned}
 (a_1, a_2, a_3, a_4, a_5) &\rightarrow (6-a_1, 6-a_2, 6-a_3, 6-a_4, 6-a_5) \\
 &\rightarrow (6-a_5, 6-a_4, 6-a_3, 6-a_2, 6-a_1) \\
 &\rightarrow (6-a_5, 6-a_4, 6-a_3, 6-a_2, 6-a_1)
 \end{aligned}$$

# 应用数据有序化

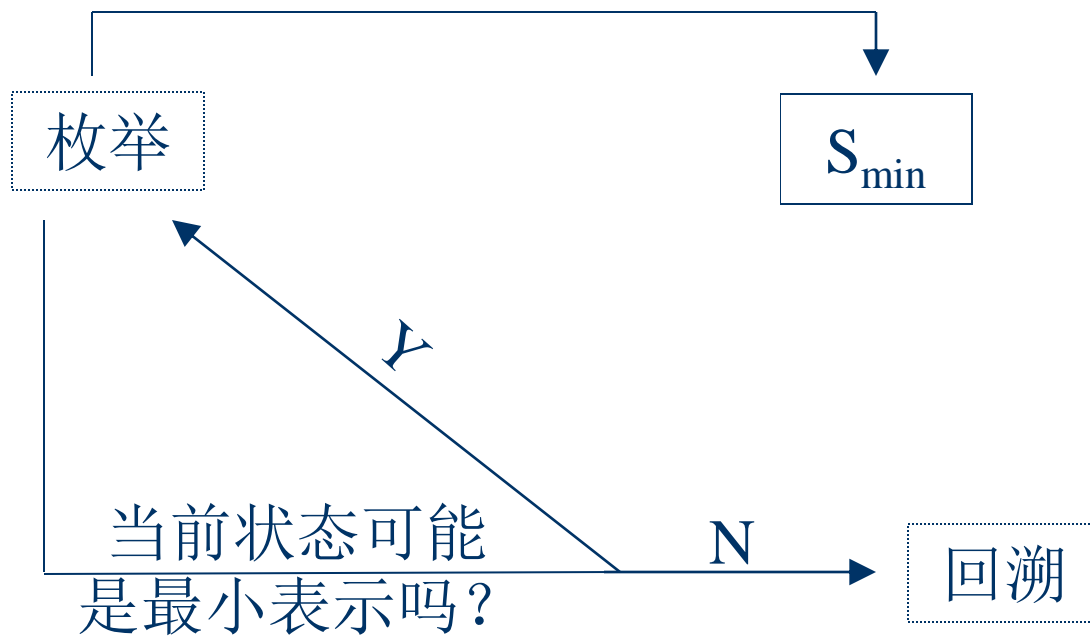




# 新的剪枝条件

$$\left\{ \begin{array}{l} a_1 \leq a_5 \\ a_1 \leq 6 - a_1 \\ a_1 \leq b_1 \\ a_1 \leq b_5 \\ a_1 \leq 6 - b_1 \\ a_1 \leq 6 - a_5 \\ a_1 \leq 6 - b_5 \end{array} \right.$$

# 空间复杂度的降低



# 应用最小表示法的算法 与常规算法的比较

	状态的生成	判断同构的费用 (或判断最小表示的费用)	时间复杂度	空间复杂度
常规算法	$O(a * N!)$ ( $a < 1$ )	$O(N *  S )$	$O(a * N! * N *  S )$	$O(N *  S )$
应用最小表示的算法	$O(b * N!)$ ( $b \ll a$ )	$O(N)$	$O(b * N! * N)$	$O(N)$



# 两种实现方法的比较

阶段	预处理	实时处理
优点	精简了数学模型	对空间的要求较低， 形式多样，应用广泛
缺点	对空间的要求较高	可能重复处理

# 总结

- ◆ 努力创造符合科学美的数据。
- ◆ 追求好的性价比。

# 谢谢！

请大家多多指教！