基于逆序数理论的排序算法严谨分析（修正版）

符号表

本文采用以下数学符号和记号：

基本符号

- $\mathbb{N}^+ = \{1, 2, 3, \ldots\}$：正整数集
- $\mathbb{R}$：实数集
- $[n] = \{1, 2, \ldots, n\}$：前 n 个正整数的集合
- $\binom{n}{k} = \frac{n!}{k!(n-k)!}$：二项式系数
- $|S|$：集合 S 的基数
- $\lceil x \rceil$：不小于 x 的最小整数
- $f(n) = O(g(n))$：大O记号，存在常数 c, n_0 使得 $f(n) \leq c \cdot g(n)$ 对所有 $n \geq n_0$ 成立
- $f(n) = \Theta(g(n))$：$f(n) = O(g(n))$ 且 $g(n) = O(f(n))$
- $f(n) = \Omega(g(n))$：存在常数 c, n_0 使得 $f(n) \geq c \cdot g(n)$ 对所有 $n \geq n_0$ 成立

序列与置换

- X_n：n 元素序列空间
- $X = (x_1, x_2, \ldots, x_n)$：长度为 n 的序列
- S_n：n 次对称群（$[n]$ 上的双射群）
- $\pi \in S_n$：置换
- $e \in S_n$：恒等置换
- $\pi \cdot X$：置换 π 作用于序列 X
- $\tau_{i,j}$：交换位置 i 和 j 的换位

逆序数相关

- $\mathrm{inv}(X)$ 或 $I(X)$：序列 X 的逆序数
- $d_\tau(X, Y)$：序列 X 和 Y 之间的 Kendall τ 距离
- $\mathrm{invcross}(L, R)$：子序列 L 和 R 之间的跨逆序对数

算法复杂度

- $T(X)$ 或 $T(n)$：算法的时间复杂度
- $T_{\mathrm{comp}}(X)$：比较次数
- $T_{\mathrm{swap}}(X)$：交换次数

- $H(X)$：序列 X 的排序信息复杂度

图论

- $\Gamma(G,S)$：群 G 关于生成集 S 的 Cayley 图
- V, E：图的顶点集和边集
- $d_G(u,v)$：图 G 中顶点 u 和 v 的距离

1. 基本定义与记号

定义 1.1（有限序列空间）设 $\mathbb{N}^+ = \{1, 2, 3, \ldots\}$，对于 $n \in \mathbb{N}^+$，定义 n 元素序列空间为：

$$X_n = \{X = (x_1, x_2, \ldots, x_n) : x_i \in \mathbb{R}, 1 \leq i \leq n\}$$

定义 1.2（对称群）n 次对称群 S_n 是集合 $\{1, 2, \ldots, n\}$ 上所有双射的集合，群运算为函数复合。

定义 1.3（置换作用）对于 $\pi \in S_n$ 和 $X = (x_1, \ldots, x_n) \in X_n$，定义置换作用：

$$\pi \cdot X = (x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(n)})$$

定义 1.4（逆序数）对于序列 $X = (x_1, x_2, \ldots, x_n) \in X_n$，定义其逆序数为：

$$\mathrm{inv}(X) = |\{(i,j) : 1 \leq i < j \leq n \text{ 且 } x_i > x_j\}|$$

示例：对于序列 $[3, 1, 2]$，逆序对为 $(1,2)$（$3 > 1$）和 $(1,3)$（$3 > 2$），因此 $\mathrm{inv}([3,1,2]) = 2$。

定义 1.5（相邻换位）对于 $1 \leq i \leq n-1$，相邻换位 $\tau_{i,i+1} \in S_n$ 定义为：

$$\tau_{i,i+1} \cdot X = (x_1, \ldots, x_{i-1}, x_{i+1}, x_i, x_{i+2}, \ldots, x_n)$$

定义 1.6（排序问题）给定序列 $X \in X_n$，排序问题是找到置换 $\pi^* \in S_n$ 使得：

$$\pi^* \in \arg\min_{\pi \in S_n} \mathrm{inv}(\pi \cdot X)$$

当 X 的元素互不相等时，解唯一存在且 $\mathrm{inv}(\pi^* \cdot X) = 0$。

2. 逆序数的基本性质

引理 2.1（逆序数的界）对于任意 $X \in X_n$：

$$0 \leq \mathrm{inv}(X) \leq \binom{n}{2} = \frac{n(n-1)}{2}$$

证明：下界显然，因为逆序对的个数非负。上界：总共有 $\binom{n}{2}$ 个位置对 (i,j) 满足 $1 \le i < j \le n$，当且仅当 $x_1 > x_2 > \cdots > x_n$（严格递减）时达到上界。□

定理 2.2 （相邻换位的逆序数效应）设 $X' = \tau_{i,i+1} \cdot X$，则：

证明：相邻换位 $\tau_{i,i+1}$ 只影响涉及位置 i 和 $i+1$ 的逆序对。具体分析：

设 $A = \{(j,k) : 1 \le j < k \le n, (j,k) \neq (i,i+1), x_j > x_k\}$，这些逆序对不受交换影响。

对于位置对 $(i,i+1)$：

- 若 $x_i > x_{i+1}$，则 $(i,i+1)$ 在 X 中是逆序对，在 X' 中不是
- 若 $x_i < x_{i+1}$，则 $(i,i+1)$ 在 X 中不是逆序对，在 X' 中是
- 若 $x_i = x_{i+1}$，则交换前后都不是逆序对

因此 $\mathrm{inv}(X') = |A| + \delta$，其中 δ 如定理所述。□

推论 2.3 任何单次相邻换位最多改变逆序数1个单位，这是所有可能交换中变化最小的。

定理 2.4 （Kendall τ 距离）对于序列 $X, Y \in X_n$，定义Kendall τ 距离：

$$d_\tau(X,Y) = |\{(i,j) : 1 \le i < j \le n \text{ 且 } (x_i - x_j)(y_i - y_j) < 0\}|$$

则 (X_n, d_τ) 构成度量空间。

证明：需验证度量公理：

1. 非负性：$d_\tau(X,Y) \ge 0$，等号当且仅当 X 和 Y 具有相同的相对顺序
2. 对称性：$d_\tau(X,Y) = d_\tau(Y,X)$ 显然成立
3. 三角不等式：对任意 X, Y, Z，有 $d_\tau(X,Z) \le d_\tau(X,Y) + d_\tau(Y,Z)$

三角不等式的证明：对于位置对 (i,j)，考虑 $(x_i - x_j)$，$(y_i - y_j)$，$(z_i - z_j)$ 的符号。若 $(x_i - x_j)(z_i - z_j) < 0$，则必有 $(x_i - x_j)(y_i - y_j) < 0$ 或 $(y_i - y_j)(z_i - z_j) < 0$ 中至少一个成立（否则三者符号关系矛盾）。□

3. 排序算法的收敛性理论

定义 3.1 （比较交换算法）比较交换排序算法是一个函数序列 $\{f_k\}_{k=0}^{\infty}$，其中每个 $f_k : X_n \to \{1, 2, \ldots, n-1\} \cup \{\perp\}$，满足：

- 若 $f_k(X) = i \neq \perp$，则执行相邻换位 $\tau_{i,i+1}$
- 若 $f_k(X) = \perp$，则算法终止

定理 3.2 （收敛性定理）设算法 $\{f_k\}$ 满足改进条件：若 $\mathrm{inv}(X) > 0$，则存在 i 使得 $f_k(X) = i$ 且 $x_i > x_{i+1}$。则算法必在有限步内收敛。

证明：设第 k 步后的序列为 $X^{(k)}$，逆序数为 $I_k = \mathrm{inv}(X^{(k)})$。

由改进条件和定理2.2，当 $I_k > 0$ 时，有 $I_{k+1} = I_k - 1 < I_k$。

序列 $\{I_k\}$ 满足：

1. $I_k \in \mathbb{N} \cup \{0\}$（离散性）
2. $I_k > 0 \Rightarrow I_{k+1} < I_k$（单调性）
3. $I_k \geq 0$（有界性）

由自然数的良序原理，严格单调递减的非负整数序列必在有限步内达到最小值0。当 $I_k = 0$ 时，序列已排序，算法终止。□

定理 3.3 （最优性下界）任何基于相邻换位的排序算法至少需要 $\mathrm{inv}(X)$ 次交换。

证明：由定理2.2，每次相邻换位最多减少1个逆序数。初始逆序数为 $\mathrm{inv}(X)$，终止时逆序数为0，因此至少需要 $\mathrm{inv}(X)$ 次交换。□

4. 置换群的组合结构

定义 4.1 （Cayley图）相邻换位生成的Cayley图 $\Gamma(S_n, T)$ 定义为：

- 顶点集：$V = S_n$
- 边集：$E = \{(\pi, \pi\tau) : \pi \in S_n, \tau \in T\}$
- 生成集：$T = \{\tau_{i,i+1} : 1 \leq i \leq n-1\}$

定理 4.2 （连通性）Cayley图 $\Gamma(S_n, T)$ 是连通图。

证明：需证明相邻换位生成整个对称群 S_n。

任意置换 $\pi \in S_n$ 可分解为不相交轮换的乘积。每个长度为 k 的轮换 $(a_1\ a_2\ \cdots\ a_k)$ 可表示为 $k-1$ 个相邻换位的乘积：

$$(a_1\ a_2\ \cdots\ a_k) = \tau_{a_1,a_2}\tau_{a_2,a_3}\cdots\tau_{a_{k-1},a_k}$$

但这里需要将任意换位表示为相邻换位。实际上，任意换位 $\tau_{i,j}$（$i < j$）可表示为：

$$\tau_{i,j} = \tau_{i,i+1}\tau_{i+1,i+2}\cdots\tau_{j-1,j}\tau_{j-2,j-1}\cdots\tau_{i,i+1}$$

因此任意置换都可表示为相邻换位的乘积。□

例：对于 S_3，其Cayley图 $\Gamma(S_3, \{\tau_{1,2}, \tau_{2,3}\})$ 的结构为：

- 顶点：$\{e, (1\ 2), (2\ 3), (1\ 3), (1\ 2\ 3), (1\ 3\ 2)\}$
- 从每个顶点出发有两条边，分别对应右乘 $\tau_{1,2}$ 和 $\tau_{2,3}$

- 图的直径为 $\binom{3}{2} = 3$，对应完全逆序 $(3,2,1)$ 到 $(1,2,3)$ 的最短路径长度

定理 4.3（测地距离）在Cayley图 $\Gamma(S_n, T)$ 中，从恒等置换 e 到置换 π 的测地距离等于 $\mathrm{inv}(\pi \cdot (1,2,\ldots,n))$。

证明：设 $X_0 = (1,2,\ldots,n)$，则 $\pi \cdot X_0$ 的逆序数即为所需的最少相邻交换次数。由定理3.3，这也是测地距离的下界。构造性地，总可以通过恰好 $\mathrm{inv}(\pi \cdot X_0)$ 次相邻交换将 $\pi \cdot X_0$ 排序为 X_0，对应从 π 到 e 的路径。$\square$

推论 4.4（直径）Cayley图 $\Gamma(S_n, T)$ 的直径为 $\binom{n}{2}$。

5. 信息论分析

定义 5.1（排序问题的信息复杂度）对于输入序列 X，其排序信息复杂度定义为：

$$H(X) = \lceil \log_2(\mathrm{inv}(X)!) \rceil$$

定理 5.2（信息论下界）任何基于比较的排序算法在最坏情况下至少需要 $H(X)$ 次比较。

证明：需要确定 $\mathrm{inv}(X)$ 个逆序对的正确顺序，这相当于在 $\mathrm{inv}(X)!$ 种可能的配置中确定正确的一种。每次比较最多提供1比特信息，因此至少需要 $\lceil \log_2(\mathrm{inv}(X)!) \rceil$ 次比较。$\square$

定理 5.3（Stirling近似下的渐近界）当 $\mathrm{inv}(X) = \Theta(n^2)$ 时：

$$H(X) = \Theta(n^2 \log n)$$

证明：使用Stirling公式 $\log(k!) = k\log k - k + O(\log k)$，得：

$$H(X) = \mathrm{inv}(X)\log(\mathrm{inv}(X)) - \mathrm{inv}(X) + O(\log(\mathrm{inv}(X)))$$

当 $\mathrm{inv}(X) = \Theta(n^2)$ 时，主导项为 $\Theta(n^2 \log n)$。$\square$

6. 典型算法的严格分析

6.1 冒泡排序

算法描述：

```
BubbleSort(X):
 for k from n-1 down to 1:
  for i from 1 to k:
   if X[i] > X[i+1]:
    swap(X[i], X[i+1])
```

定理 6.1（冒泡排序的交换复杂度）冒泡排序的交换次数恰好等于输入的逆序数：

$$T_{\text{swap}}(X) = \text{inv}(X)$$

证明：算法只对满足 $X[i] > X[i+1]$ 的相邻对执行交换。由定理2.2，每次交换使逆序数减少1。算法终止时逆序数为0，因此总交换次数等于初始逆序数。□

定理 6.2 （冒泡排序的比较复杂度）冒泡排序的比较次数为：

$$T_{\text{comp}}(X) = \binom{n}{2} = \frac{n(n-1)}{2}$$

与输入无关。

证明：外层循环执行 $n-1$ 次，第 k 次内层循环执行 k 次比较。总比较次数：

6.2 插入排序

算法描述：

```
InsertionSort(X):
 for k from 2 to n:
  j = k
  while j > 1 and X[j-1] > X[j]:
   swap(X[j-1], X[j])
   j = j - 1
```

定理 6.3 （插入排序的复杂度）插入排序的时间复杂度为：

$$T(X) = n - 1 + \text{inv}(X)$$

其中 $n-1$ 是比较次数的下界，$\text{inv}(X)$ 是交换次数。

证明：处理第 k 个元素时：

- 比较次数：$1 + |\{j : j < k, X[j] > X[k]\}|$
- 交换次数：$|\{j : j < k, X[j] > X[k]\}|$

总交换次数 = $\sum_{k=2}^{n} |\{j : j < k, X[j] > X[k]\}| = \text{inv}(X)$

总比较次数 = $\sum_{k=2}^{n} (1 + |\{j : j < k, X[j] > X[k]\}|) = (n-1) + \text{inv}(X)$ □

6.3 归并排序

定理 6.4 （归并排序的逆序数分解）对于序列 $X = L \oplus R$（左右子序列的连接），有：

$$\mathrm{inv}(X) = \mathrm{inv}(L) + \mathrm{inv}(R) + \mathrm{invcross}(L, R)$$

其中 $\mathrm{invcross}(L, R) = |\{(i, j) : i \in L, j \in R, x_i > x_j\}|$。

定理 6.5 （归并排序的时间复杂度）归并排序的时间复杂度为：

$$T(n) = \Theta(n \log n)$$

与输入的逆序数分布无关。

证明：递推关系 $T(n) = 2T(n/2) + \Theta(n)$，由主定理得 $T(n) = \Theta(n \log n)$。□

6.4 选择排序

算法描述：

```
SelectionSort(X):
 for k from 1 to n-1:
  min_idx = k
  for j from k+1 to n:
   if X[j] < X[min_idx]:
    min_idx = j
  swap(X[k], X[min_idx])
```

定理 6.6 （选择排序的非自适应性）选择排序的时间复杂度为：$T(X) = \Theta(n^2)$，与输入序列的逆序数无关。

证明：算法的两层循环结构决定了：

- 外层循环执行 $n - 1$ 次
- 第 k 次外层循环时，内层循环执行 $n - k$ 次比较

总比较次数：

$$\sum_{k=1}^{n-1} (n - k) = \sum_{i=1}^{n-1} i = \frac{n(n-1)}{2} = \Theta(n^2)$$

选择排序每轮都必须扫描剩余元素找到最小值，无法利用输入的有序性，因此是完全非自适应的。□

定理 6.7 （选择排序的逆序数效应）选择排序第 k 轮将最小元素 x 交换到位置 k 时，消除的逆序数为：$\Delta I_k = |\{j : j > k, X[j] < x\}|$

该式理论上表示若将最小元素 x 移动到位置 k 所消除的逆序数，但由于 x 是最小值，右侧没有更小的元素，故实际值为0。选择排序的交换操作不直接反映逆序数的减少，其非自适应性正在于此。□

6.5 快速排序

算法描述：

```
QuickSort(X, low, high):
 if low < high:
  pivot_pos = Partition(X, low, high)
  QuickSort(X, low, pivot_pos - 1)
  QuickSort(X, pivot_pos + 1, high)

Partition(X, low, high):
 pivot = X[high]
 i = low - 1
 for j from low to high - 1:
  if X[j] <= pivot: // 修正：使用 j 而非 i
   i = i + 1
   swap(X[i], X[j])
 swap(X[i + 1], X[high])
 return i + 1
```

定理 6.8 （快速排序的期望复杂度）在随机选择主元的情况下，快速排序的期望时间复杂度为：

$$E[T(X)] = \Theta(n \log n)$$

定理 6.9 （快速排序与逆序数的关系）快速排序的性能与输入的逆序数分布存在复杂关系：

1. **最好情况**：当每次分割都接近均匀时，$T(X) = \Theta(n \log n)$
2. **最坏情况**：当输入已排序或逆序时，$T(X) = \Theta(n^2)$
3. **平均情况**：主元选择的随机性使得期望复杂度为 $\Theta(n \log n)$

证明框架：

- 最好情况：每次分割将问题规模减半，递推关系 $T(n) = 2T(n/2) + \Theta(n)$
- 最坏情况：每次分割极不均匀，递推关系 $T(n) = T(n-1) + \Theta(n)$，解得 $T(n) = \Theta(n^2)$
- 平均情况：需要分析所有可能分割的概率分布，通过概率论方法可证明期望复杂度为 $\Theta(n \log n)$ □

注记：虽然快速排序不是直接的逆序数自适应算法，但其性能确实受到输入数据分布的影响。在某些特殊的逆序数分布下（如接近有序），快速排序可能表现较差。

7. 复杂度比较与最优性

定理 7.1 （算法复杂度比较）各算法的复杂度总结：

算法	时间复杂度	空间复杂度	稳定性	自适应性
冒泡排序	$O(n^2)$	$O(1)$	稳定	部分自适应
插入排序	$O(n + \text{inv}(X))$	$O(1)$	稳定	完全自适应
选择排序	$\Theta(n^2)$	$O(1)$	不稳定	非自适应
归并排序	$\Theta(n \log n)$	$O(n)$	稳定	非自适应
快速排序	期望 $O(n \log n)$，最坏 $O(n^2)$	期望 $O(\log n)$，最坏 $O(n)$	不稳定	间接相关

注：快速排序的空间复杂度主要指递归栈深度。

定理 7.2 （最优性定理）在基于比较的排序算法中：

1. **最坏情况下界**：$\Omega(n \log n)$
2. **插入排序在 $\text{inv}(X) = O(n)$ 时达到 $O(n)$ 最优复杂度**
3. **归并排序在最坏情况下达到渐近最优**

证明：

1. 信息论下界：排序 n 个不同元素需要区分 $n!$ 种排列，每次比较提供最多1比特信息，因此至少需要 $\lceil \log_2(n!) \rceil = \Omega(n \log n)$ 次比较。
2. 当输入接近有序时（$\text{inv}(X) = O(n)$），插入排序的复杂度为 $O(n + \text{inv}(X)) = O(n)$，达到线性最优。
3. 归并排序在任何输入下都保持 $\Theta(n \log n)$，匹配最坏情况下界。

8. 进阶理论分析

8.1 逆序数的概率分布

定理 8.1 （随机序列的逆序数期望）对于随机排列 $\pi \in S_n$，其逆序数的期望值为：$E[\text{inv}(\pi)] = \frac{n(n-1)}{4}$

证明：对于任意位置对 (i, j)，$i < j$，有 $P(x_i > x_j) = \frac{1}{2}$（假设元素值连续且无重复）。因此：$E[\text{inv}(\pi)] = \sum_{1 \le i < j \le n} P(x_i > x_j) = \binom{n}{2} \cdot \frac{1}{2} = \frac{n(n-1)}{4}$ $\square$

定理 8.2 （逆序数的方差）随机排列的逆序数方差为：$\text{Var}[\text{inv}(\pi)] = \frac{n(n-1)(2n+5)}{72}$

8.2 平均情况复杂度

推论 8.3 （平均复杂度）各算法在随机输入下的平均时间复杂度：

- **冒泡排序**：$E[T] = \frac{n(n-1)}{4} + \frac{n(n-1)}{2} = \frac{3n(n-1)}{4} = \Theta(n^2)$
- **插入排序**：$E[T] = (n-1) + \frac{n(n-1)}{4} = \frac{n(n+3)}{4} - 1 = \Theta(n^2)$
- **归并排序**：$E[T] = \Theta(n \log n)$（与输入无关）

8.3 自适应性的定量分析

定义 8.1 （自适应比）算法 A 的自适应比定义为：$\rho_A = \max_{X \in X_n} \frac{T_A(X)}{T_A^{\text{opt}}(X)}$

其中 $T_A^{\text{opt}}(X)$ 是算法 A 在输入 X 上的理论最优复杂度。

定理 8.4 （自适应比分析）

- **插入排序**：$\rho_{\text{ins}} = 1$（完全自适应）
- **冒泡排序**：$\rho_{\text{bubble}} = \frac{n-1}{\text{inv}(X)} + 1$（部分自适应）
- **选择排序**：$\rho_{\text{select}} = \infty$（非自适应）

9. 实际应用与优化策略

9.1 混合排序算法

算法 9.1 （Introsort策略）

```
IntroSort(X, depth_limit):
 if |X| <= INSERTION_THRESHOLD:
  InsertionSort(X)
 elif depth_limit == 0:
  HeapSort(X)
 else:
  pivot_pos = Partition(X)
  IntroSort(X[1:pivot_pos-1], depth_limit - 1)
  IntroSort(X[pivot_pos+1:|X|], depth_limit - 1)
```

其中 `INSERTION_THRESHOLD ≈ 16`，`depth_limit = 2⌊log₂ n⌋`。

定理 9.1 Introsort 的时间复杂度为 $O(n \log n)$，结合了快速排序的平均性能和堆排序的最坏情况保证。

9.2 并行排序算法

定理 9.2 （并行归并排序）在 p 处理器的并行模型下，归并排序的时间复杂度为：$T_p(n) = O\left(\frac{n \log n}{p} + \log n\right)$

证明框架：

- 分治阶段可以完全并行化，深度仍为 $O(\log n)$
- 每层的工作量 $O(n)$ 可在 p 个处理器间均匀分配
- 通信开销主要在合并阶段

结论

本文基于逆序数理论对经典排序算法进行了严谨的数学分析，建立了统一的理论框架，给出了各主要算法的精确复杂度界，并通过 Cayley 图理论提供了排序问题的几何直观。逆序数作为衡量序列"无序程

度"的核心指标，为理解和优化排序算法提供了深刻的数学基础。

参考文献

[1] Knuth, D. E. *The Art of Computer Programming, Volume 3: Sorting and Searching*. 2nd Edition, Addison-Wesley, 1998.

[2] Sedgewick, R. *Algorithms in C++: Fundamentals, Data Structures, Sorting, Searching*. 3rd Edition, Addison-Wesley, 2001.

[3] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. *Introduction to Algorithms*. 3rd Edition, MIT Press, 2009.