arXiv:2611.30206v1 [cs.DS] 23 Nov 2026

位置感知匹配统计

LorraineA. K. Ayad [1]、GrigoriosLoukides [2] 和 Solon P.Pissis [3]

[1]伦敦布鲁内尔大学，英国[2]伦敦国王学院，英国[3]塞浦路斯研究所，尼科西亚，塞浦路斯

摘要

编辑距离是衡量字符串相似度的标准指标，但其二次时间复杂度使其不适用于长字符串。流行的线性时间替代方法，例如 q-gram 距离，或基于匹配统计量（MS）的平均公共子串度量（ACS），会考虑两个输入字符串中共有的子串，却不考虑它们在各字符串中的位置。然而，在许多实际应用中，例如比较文档版本或共同记录的时间序列，只有当共有子串的出现位置相近时才真正相关。

我们介绍位置感知匹配统计（LAMAS）。给定字符串 X 和 Y 以及位移容差 d，LAMAS 对于 Y的每个位置 i，报告从 i 开始的最长子串；该子串出现在 X 中的位置 j，且满足 $|j-i| \leq d$，同时给出位移（$j-i$）。当 d 无界时，MS 是 LAMAS 的特殊情形。我们的主要成果是一种算法：对于长度至多为 n 的字符串和任意 d，在 O(n) 时间和空间内求解 LAMAS；这已达到最优，并且与 MS 的复杂度相匹配。该算法将 LAMAS 归约为滑动窗口上的后缀秩前驱与后继查询，并通过窗口分解和区间并查集在线性总时间内回答这些查询。

在 20 个真实数据集和合成数据上的实验表明，在基于相似度的分类、top-K 检索和聚类方面，LAMAS 的表现显著优于 MS 和 q-gram 距离。在分类方面，它的表现可与编辑距离及一种受约束的动态时间规整变体相媲美；在基于编辑距离的 top-K 检索方面，它总体上优于一种最先进的方法。最后，我们实现的 LAMAS 主要算法具有线性的 n 扩展性，并且不会随着 d增大而变慢。

1 引言

字符串（即字母表上的元素序列）是许多领域中的一种基础数据类型，应用范围从数据库 [SKS20] 和信息检索 [BR11]到生物信息学 [Gus97]。例如，时间序列通常会离散化为由小型字母表构成的字符串 [PKS25，LKWL07], 而 DNA序列则自然地表示为由四种核苷酸构成的字母表上的字符串。衡量两个字符串的相似度是许多现实应用的核心，例如 top-K 相似性搜索 [ZZ20a，LPS23],记录链接 [CRS20], 聚类 [Yan14], 和分类 [MRS08]。

或许，比较两个字符串最常见的方法是计算它们的编辑距离（ED），也称为莱文斯坦距离 [Lev66]：将一个字符串转换为另一个字符串所需的最少字符插入、删除和替换次数。对于长度均不超过 n的两个字符串，经典的动态规划算法可在$O(n^2)$ 时间 [WF74]内计算它们的 ED。不幸的是，这一上界仅通过多对数因子得到改进 [CLZ03、BF08、Gra16], ，而条件下界表明，不存在真正的次二次算法 [BI18、ABW15、BK15、AHWW16]。

为绕过这一二次障碍，人们提出了许多可高效计算的度量，用于比较两个字符串中的组成模式，分别记作 X和 Y。广泛使用的 q-gram 距离 [Ukk92] 比较 X和 Y中长度为q-子串的出现频率，而平均公共子串（ACS）度量则对每个字符串的所有位置取平均，计算从该位置开始、且也出现在另一个字符串中的最长子串长度 [UBTC06]。与 ED 不同，这两种度量都能在 O(n) 时间内计算（参见 [Ukk92、CL90]）。然而，它们只能捕捉两个字符串共享哪些模式，却无法体现这些模式出现的位置：无论 X 中的某次出现距离 Y 的对应位置近还是远，对 ACS 的贡献都相同。

1.1 动机

在许多应用中，字符串中的位置 X 和 Y 都可以相互比较，共享模式的重要性取决于其出现的位置。例如，在比较文档的两个版本以识别未更改的内容时 [MM85], ，出现在大致相同位置的共享短语，比只出现在无关章节中的短语更有参考价值。然而，要求位置完全一致过于严格，因为插入和删除会使后续文本发生偏移。

将时间序列离散化为字符串时，也会出现类似问题。考虑在同一时期记录并编码为字符串 X和 Y的两条股价序列。两条股价序列中共享的涨跌模式，若出现在大致相同的时间，则更能说明它们的走势相似。事实上，在时间序列匹配中，允许有限的时间偏移早已是成熟做法。例如，著名的动态时间规整（DTW） [Keo02] 度量会对齐不同位置的观测值，而带状动态时间规整则将对齐路径限制在以对角线为中心、半径为 d 的带状区域内 [SC78], ，类似于对角带状编辑距离 [Ukk85]。与编辑距离（ED）一样，教科书中介绍的 DTW [SC78] 的 O(n^2) 时间算法仅通过次对数因子得到改进 [GS18] ，并且不太可能实现显著改进 [ABW15, BK15],；而带状动态时间规整的时间复杂度为 O(nd)。位置约束也应用于时间序列数据库查询 [FKLR05] 以及字符串索引：模式搜索被限制在指定的文本区间内 [MN06]。

以下经验性观察是许多计算编辑距离的算法的理论基础，也证明了上述直觉的合理性；对齐的定义请参见第 3 节。

引理1.1 ([Ukk85])。 任取 X 与 Y 的一个比对，其代价至多为 k。若 $X[j]$ 与 $Y[i]$ 对齐（作为匹配或替换），则 $|j - i| \le k$。

这引出了本研究的核心问题：

> 我们能否设计一种基于共享模式的字符串相似度度量，既考虑这些模式的位置，又能在线性时间内计算？

1.2 我们的方法

为回答这个问题，我们基于匹配统计原语 [CL90、Gus97、LMP26]，（它是 ACS 的基础），而非 q-元组，因为 MS不固定模式的长度：它会针对 Y的每个位置最大化匹配长度，从而捕获任意长的匹配。形式化地，给定两个字符串 $X = X[0..|X|)$ 和 $Y = Y[0..|Y|)$，Y 相对于 X 的匹配统计量是数组$\mathsf{MS}_X = \mathsf{MS}_X[0..|Y|)$，其定义对所有 $i \in [0..|Y|)$ 均成立，具体如下：

$$\mathsf{MS}_X[i] = \max_{0 \le j < |X|} \mathrm{LCP}\big(X[j..|X|), Y[i..|Y|)\big), \tag{1}$$

其中 LCP（U，V）表示两个字符串 U和 V的最长公共前缀（LCP）长度。直观地说，$\mathsf{MS}_X[i]$ 是 Y 中从位置 i开始、且在 任意位置出现在 X中的最长子串的长度。数组 MS_Y的定义与此类似。两个数组都可以在 O（$|X| + |Y|$）时间内由 [CL90]计算。对于 X 和 Y，ACS 度量定义如下：[1]

$$\mathrm{ACS}(X,Y) = \frac{1}{2}\Big(\frac{1}{|Y|}\sum_{i=0}^{|Y|-1} \mathsf{MS}_X[i] + \frac{1}{|X|}\sum_{i=0}^{|X|-1} \mathsf{MS}_Y[i]\Big).$$

在方程（ j 1）中不受限制地选择 j，可能会选择较短的近距离匹配，这并不适用于两个字符串中位置相近的情况。另外，简单地从 MS 的输出中丢弃远距离匹配，也无法恢复最佳的近距离匹配，因为较短的备选匹配并未被报告。

因此，我们引入位置感知匹配统计（LAMAS），将每个匹配的起始位置限制在查询位置的位移容差 d 以内：对于 Y的每个位置 i，LAMAS 找出 Y 中从位置 i开始、且在 X 中从满足 $|j - i| \le d$ 的位置 j开始出现的最长子串，并返回其长度 l_i和位移 $\delta_i = j - i$。

问题 1 (LAMAS)。给定两个字符串 X和 Y 其长度均不超过 n，以及一个整数 $d \ge 0$，对于每个 $i \in [0..|Y|)$，输出一个数对 (l_i, δ_i)，其中

$$\ell_i = \max_{\substack{0 \le j < |X| \\ |j-i| \le d}} \mathrm{LCP}\big(X[j..|X|), Y[i..|Y|)\big)$$

并且 $\delta_i = j - i$ 对应于取得该最大值的位置 j （若有多个，则任意选择）。如果没有位置$j \in [0..|X|)$满足 $|j - i| \le d$，则 $l_i = 0$和 δ_i未定义。

至于 MS，我们用 LMS_X 表示长度数组，即 $\mathsf{LMS}_X[i] = l_i$，并以类似方式定义 LMS_Y ly.

示例 *1.2*。设 $X = abcdef$，$Y = defabc$。$Y[0..6)$ 的最长前缀在 X 中出现为 def，位置为 $j = 3$，因此 $MS_X[0] = 3$；类似地，$MS_X = MS_Y = (3, 2, 1, 3, 2, 1)$。对于 $d = 2$ 的 $LAMAS$，$i = 0$ 的可允许位置为 $j \in \{0, 1, 2\}$，这些位置的起始字符都不是 $Y[0] = d$，因此 $LMS_X[0] = 0$。同样，Y 的每个字母在 X 中都只在位移 ± 3 处出现，因此 $LMS_X = (0, 0, 0, 0, 0, 0)$，而且对于每个 i，$(l_i, \delta_i) = (0, 0)$ 都是有效输出。

容差 d 可以根据领域知识来设定。对于时间序列，它指对应事件之间可接受的最大延迟，例如带状动态时间规整的扭曲窗口 [SC78]。如果预期字符串之间的编辑距离（ED）不超过 k，就像关系密切的 DNA序列那样，设定 $d = k$ 可确保比对中代价不超过 k 的每个精确匹配都从允许的位置开始（引理 1.1）。如果没有这类知识，d 可以根据数据样本进行估计（例如，根据抽取的若干样本对的编辑距离），也可以像我们的实验那样，通过带标签的验证数据进行调优（见第 5.2节）。

例1.3 通过真实股票价格展示了价格变动的时机为何重要 [Mac97], 忽略位置可能会颠倒相似度排名，而 LAMAS 可以纠正这一点。

[1]为简洁起见，我们省略了对数 校正项，公式为 [UB TC06]。



图1：股价时间序列中的共享模式。

例*1.3*。我们使用 *SAX [LKLC03]*，将苹果公司、微软和莫德纳的股票从*2*月*3*日到六月*30*日 *2020 [Yah20]* 的每日调整收盘价离散化为长度为 $n = 20$、由 $\sigma = 4$个字母组成的字符串，使每个字母大约对应一个交易周（图*1*）。苹果公司和莫德纳具有相同的长模式 *abbcccdddd*，但它们出现的位置相差三位（约三周）；而苹果公司和微软则在相同位置具有相同的较短模式 *cccdddd*。因此，*ACS* 将莫德纳与苹果公司的相似度排在微软之前（*4.35* 对 *3.85*）。在 *ACS* 中用 *LAMAS* 替换 *MS*（$d = 1$）后，会排除错位的匹配，并颠倒排名（*1.425* 对 *3.1*）；在 $d = 0$ 或 $d = 2$使用 *LAMAS* 也会得到同样结果。这个排名与每日对数收益率的皮尔逊相关系数一致；后者是衡量股票联动的成熟指标 *[PGR+02]*：苹果公司与微软的相关系数为 *0.9374*，而苹果公司与莫德纳的相关系数仅为 *0.0027*。

例1.4 表明，基于 MS 的 ACS 可能会将两个 ED 达到最大值的字符串评为高度相似，而 LAMAS 在 d 排除相距较远的匹配时则不会如此。

例1.4. 设 n 能被3整除，并考虑

$$X = \mathsf{a}^{n/3}\mathsf{c}^{n/3}\mathsf{g}^{n/3} \quad \textit{and} \quad Y = \mathsf{g}^{n/3}\mathsf{t}^{n/3}\mathsf{a}^{n/3},$$

其中，U^k 表示将字符串 U连接 k 份所得的字符串。X 和 Y 的 ED 为 n，这是可能达到的最大值。尽管如此，*MS* 仍匹配了共享块$\mathsf{a}^{n/3}$ 和$\mathsf{g}^{n/3}$，因此 *ACS* 将 X 和 Y 评为高度相似。相比之下，所有共享字母的位置彼此相距都超过 $n/3$ 个位置，因此对于每个 $d \le n/3$，都有$LMS_X[i] = LMS_Y[i] = 0$对所有 i均成立，所以 *LAMAS* 完全没有报告相似性，这与 *ED* 的结果一致。

1.3 贡献

除了提出 LAMAS 问题外，我们的工作还作出了以下贡献。

LAMAS 的时间最优算法。标准字符串索引支持最长公共扩展（LCE）查询，经过 $O(1)$ 时间的 $O(n)$时间预处理后 [KSB06、KLA+01、BF00]。因此，对一个字符串中的每个起始位置，检查另一个字符串中的所有 $O(d)$ 个可允许位置来计算 LAMAS，总共需要 $O(nd)$ 时间。我们的核心贡献是提出一种 $O(n)$ 时间、占用 $O(n)$空间的 LAMAS 算法。

我们观察到，对于一个字符串中的每个起始位置，只需检查另一个字符串中至多两个可允许后缀：在查询后缀 $Y[i\,..\,|Y|]$两侧按字典序最近的可允许后缀（形式上，即在起始位置为 $j$with $|j - i| \le d$ 的 X 字符串的后缀中，$Y[i\,..\,|Y|]$的前驱和后继）。随着可允许窗口滑动，在平衡搜索树中维护这些后缀的秩，即可得到一个简单的 O $(n \log d)$ 时间算法。我们通过以下方式改进该算法：

- 缩小搜索空间。每组相邻查询只涉及全局排序后缀中的一小部分。我们利用这种局部性，将查询划分为多个块，并用局部排名替换全局后缀排名；这样既能保留答案，又能缩小搜索域。由此可实现 O $(n \log\log d)$ 时间复杂度；当 $d = \log_{O(1)} n$ 时，时间复杂度为 O (n)。

- 利用窗口结构。连续的查询窗口有大量重叠，且变化规律可预测。将每个窗口分解为一个固定部分和两个在合适的查询顺序下逐渐缩小的部分，就能用有序扫描和区间并查集 [GT85], 处理所有查询，总时间为$O(n)$，且与 d无关。

因此，我们实现了最优的 O(n) 时间复杂度和 O(n) 空间复杂度，与经典 MS 的渐近复杂度相匹配，尽管增加了位移约束，并对我们的核心问题给出了肯定的回答。

实验评估。 我们在 20 个真实数据集和合成数据上评估 LAMAS，并将其与 MS、 q-gram 距离、编辑距离、带状动态时间规整 [SC78], 和 MinSearch 进行比较 [ZZ20a], ——一种最先进的、用于在编辑距离下进行 top-K 检索的方法。我们考察三种常见的基于相似度的任务：

(1) 配对分类。 在测试的全部 15 个数据集上，LAMAS 的表现均优于 MS，平均 F_1 值为 0。7644，分别比 MS 和 q-gram 距离高 9。2% 和 8。7%。此外，其有效性与编辑距离、带状动态时间规整 [SC78] 相当。

(2) *Top-K* 检索。 在测试的全部 15 个数据集上，LAMAS 的表现均优于 MS 和 q-gram 距离；其平均 F_1@10 [MRS08] 为 0。7516；分别比 MS、。9%、57% 和 15% 高于 q-gram 距离和 MinSearch。此外，对于 $K \geq 5$，其平均检索有效性也达到最高。(3) 聚类。 在两个真实数据集上，LAMAS 在四项质量指标上的表现均优于 MS 和 q-gram 距离；调整兰德指数 [HA85] 分别最高提升 146% 和 154%。在调整兰德指数方面，其表现也比编辑距离最高提升 96%。

效率。 我们的 LAMAS 算法在 DNA 测序数据集上的运行时间与 MS 相近，且占用内存更少。它在 n 中还能线性扩展，并且不会随着 d 增大而变慢。

1.4 论文结构

第 2 节讨论相关工作，第3节介绍预备知识。第4节介绍我们的 LAMAS 算法，第5节介绍实验。第 6 节对论文进行总结。

2 相关工作

经典字符串相似度度量。 经典字符串相似度度量包括汉明距离、编辑距离、最长公共子序列（LCS）及其变体；综述请参见 [Nav01] 。汉明距离计算位置不匹配的数量，而编辑距离允许插入、删除和替换。Damerau–Levenshtein 距离还允许相邻转置 [Dam64]。最长公共子序列仅允许插入和删除（间隙） [Hir75], , 而最长公共子串 [CKRP26] 要求连续性。与这些度量不同，LAMAS 在每个位置记录有界位移范围内的最长精确匹配，从而为基于编辑距离的相似度提供一种高效、具备位置感知能力的代理度量。

编辑距离嵌入与学习得到的相似度。 度量嵌入将字符串映射到汉明空间，同时近似保持编辑距离 [CGK16、CGKK18]。EmbedJoin 利用此类嵌入进行编辑相似度连接 [ZZ17]。基于学习的方法则学习编辑代价 [RY98] 或相似度表示 [DYZ+20，LF22，LWYH25]。与这些方法不同，LAMAS 无需训练；而且与（随机化）嵌入不同，它是确定性的；但它不提供形式化的编辑距离近似保证。

无比对字符串度量。无比对度量通过共享或缺失的模式来比较字符串。 q-gram 距离比较子串频率 [Ukk92], , 其变体阈值q-gram 距离则将出现次数设定上限，以减少空间占用 [CFLM25]。 q-gram 距离已用于多种生物信息学工具，例如 [GIM+16、AP17]。最小缺失词（MAW）度量比较缺失的模式 [CC12, CCF+18]。匹配统计量（MS） [CL94] 是一种广泛使用的原语；近期综述请参见[LMP26] 。它也是诸如 [ARK+21、ROL+22]等软件工具的基础。LAMAS 通过限制匹配起始位置之间的位移来推广 MS；正如我们的实验所示，这有助于多种应用，而且其计算所需的时间复杂度与 MS 相同（且为最优）。

时间序列的距离。 时间序列可以使用弹性度量进行比较，例如动态时间规整（DTW） [Keo02], 和最长公共子序列（LCSS） [VGK02]；综述请参见 [HMB24] 。保序匹配统计量（OPMS）则捕捉相对值顺序上的相似性，从而支持基于趋势的比较 [LZL+26]；这与 LAMAS 互为正交。动态子序列扭曲（DSSW）比较提取出的数值特征 [ZHD26]。另一种流行的时间序列比较方法是先将其离散化为字符串，例如使用 SAX（符号聚合近似） [LKWL07，LKLC03]。这能降低维度和存储需求，同时支持基于字符串的处理。LAMAS 处理符号序列；与这些通常需要二次时间计算的弹性度量不同，LAMAS 可在最优的$O(n)$ 时间内计算。

3 预备知识

计算模型。 本文始终假设采用标准的字 RAM 模型，机器字长为 $w = \Omega(\log n)$ 位，其中 n 是输入的总大小。

字符串。对于整数 i 和 j，我们将 $[i..j]$ 写作 $\{i,..,j\}$，将 $[i..j)$ 写作 $\{i,..,j-1\}$；当 $i > j$ 时，两者均为空；当 $i = j$ 时， $[i..j)$ 也为空。字母表 Σ 是一个有限的全序集合，其元素称为字母。我们假设存在整数字母表 $\Sigma = [0..\sigma)$，且 $\sigma = n^{O(1)}$，因此字母可以容纳在机器字中。字符串 $X = X[0]X[1]\cdots X[|X|-1]$ 是由 $|X|$ 个来自 Σ 的字母组成的序列，其中 $|X|$ 是其长度；空字符串 ε 的长度为 0。对于 $0 \le i \le j \le |X|$，片段 $X[i..j)$ 是字符串 $X[i]X[i+1]\cdots X[j-1]$（当 $i = j$ 时为空），我们将 $X[i..j]$ 写作 $X[i..j+1)$。片段 $X[0..j)$ 是 X 的前缀，而片段 $X[i..|X|)$ 是 X 的后缀；如果前缀或后缀不同于 X，则称其为真前缀或真后缀。两个字符串 U 和 V 的连接记作 UV。如果 $W = X[i..i+|W|)$，则称字符串 W 在 X 的位置 i 处出现。

字典序和最长公共前缀。两个字符串 U 和 V 的最长公共前缀（LCP）是同时作为二者前缀的最长字符串；我们用 $\mathrm{LCP}(U,V)$ 表示其长度。字符串上的字典序 $\prec$ 定义为：当且仅当 U 是 V 的真前缀，或 $\mathrm{l} = \mathrm{LCP}(U,V) < \min(|U|,|V|)$ 且 $U[\mathrm{l}] < V[\mathrm{l}]$ 时成立。我们记 $U \preceq V$，当 $U \prec V$ 或 $U = V$ 时成立。我们将广泛使用事实 3.1。

事实 3.1 ([MM93])。 设 $U \preceq V \preceq W$ 为字符串。则 $\mathrm{LCP}(U, W) = \min\{\mathrm{LCP}(U, V), \mathrm{LCP}(V, W)\}$。

编辑距离与比对。两个字符串 X 和 Y 的比对是一个序列 $((u_1,v_1),\ldots,(u_z,v_z))$，由列 $(u_t,v_t) \in (\Sigma \cup \{\varepsilon\})^2 \setminus \{(\varepsilon,\varepsilon)\}$组成，满足 $u_1u_2\cdots u_z = X$和 $v_1v_2\cdots v_z = Y$。如果 $u_t = v_t$，则列 (u_t,v_t) 是匹配；如果 $u_t \neq v_t$且 $u_t, v_t \in \Sigma$，则是替换；如果 $v_t = \varepsilon$，则是删除；如果 $u_t = \varepsilon$，则是插入。如果 (u_t,v_t) 是匹配或替换，且 $u_t = X[j]$和 $v_t = Y[i]$，我们称$X[j]$与 $Y[i]$对齐，并称 $j-i$为该列的位移。在单位代价下，比对的代价等于其非匹配列的数量；编辑距离（或莱文斯坦距离 [Lev66]）$\mathrm{ED}(X,Y)$ 则是 X和 Y的比对的最小代价。

q-gram 距离。 q-gram 距离 [Ukk92] 是l_1 长度为 q 的所有子串在 X 和 Y的出现次数向量之间的距离。它可以在线性时间内计算 [Ukk92]。

后缀数组、逆后缀数组和 LCP数组。设 S 是一个长度为 m 、基于整数字母表的字符串。 S 的所有后缀均不相同，因此 S 的后缀数组是 $_S[0..m)$ 的唯一排列 SA，使得 $\mathrm{SA}_S[r]$ 是 S 按字典序排列后的第 r 小后缀的起始位置。逆后缀数组 $\mathrm{iSA}_S[0..m)$ 定义为 $\mathrm{iSA}_S[\mathrm{SA}_S[r]] = r$；因此，$\mathrm{iSA}_S[i]$ 是后缀 $S[i..m)$ 在 S 的所有后缀中的秩，并且 $S[i..m) \prec S[j..m)$ 当且仅当 $\mathrm{iSA}_S[i] < \mathrm{iSA}_S[j]$。

LCP数组 *LCP*数组 $\mathrm{LCP}_S[0..m)$ 存储按字典序相邻后缀的 LCP 长度：$\mathrm{LCP}_S[0] = 0$ 以及

$$\mathrm{LCP}_S[r] = \mathrm{LCP}\big(S[\mathrm{SA}_S[r-1]..m), S[\mathrm{SA}_S[r]..m)\big),$$

对于 $r \in [1..m)$。长度为 m、字母表大小为 $m\mathrm{O}(1)$ 的整数字母表上的字符串，其后缀数组可在 O (m) 时间内构造 [KSB06]；随后可在 O (m) 时间内得到逆后缀数组，并可根据 S、SA_S 和 iSA_S [KLA+01] 在 O (m) 时间内计算 LCP数组。在第 4 节中，我们为字符串 $S = X\#Y$构造这些数组，其中 $\# \notin \Sigma$ 是哨兵字符，我们假设它按字典序大于 Σ 中的每个字符；因此， S 的字母表为 $[0..\sigma]$，其中包含 $\# = \sigma$，以及 $|S| = |X| + |Y| + 1$。从位置 $|X|$ 开始的 S 的后缀，是唯一一个以 # 开头的后缀；S 的其余每个后缀，要么从 X 开始（位置$[0..|X|)$），要么从 Y 开始（位置 $[|X|+1..|S|)$）。

区间最小值查询。令 $A[0..m)$ 为一个数组。对于$0 \le i \le j < m$, 区间最小值查询 $\mathrm{RMQ}_A(i,j)$ 返回一个索引 $t \in [i..j]$，使得 $A[t] = \min A[i..j]$（并列时任意选择）。对 A 进行 O(m) 时间的预处理后，任何区间最小值查询（RMQ）都可以在 O(1) 时间内回答 [BF00]。

最长公共扩展查询。对长度为 m 的字符串 S 进行最长公共扩展（LCE）查询时，给定两个位置 i, $j \in [0..m)$，查询值 $\mathrm{LCE}_S(i,j) = \mathrm{LCP}(S[i..m), S[j..m))$。将事实 3.1 应用于排序后的后缀序列，可将 LCE 查询归约为对 LCP数组的区间最小值查询：对于$i \ne j$，令 $r = \mathrm{iSA}_S[i]$和 $s = \mathrm{iSA}_S[j]$，不失一般性，

$$\mathrm{LCE}_S(i,j) = \min \mathrm{LCP}_S[r+1..s] = \mathrm{LCP}_S[t^*],$$

其中 $t^* = \mathrm{RMQ}_{\mathrm{LCP}_S}(r+1, s)$，而 $\mathrm{LCE}_S(i, i) = m - i$。因此，对 S（后缀数组、逆后缀数组、LCP数组以及其上的 RMQ结构）进行 O(m) 时间的预处理后，每个 LCE查询都能在 O(1) 时间内回答；我们将这一组合称为 S 的 *LCE* 数据结构 。对于 $S = X\#Y$，LCE 数据结构还可以回答 X 的一个后缀与 Y 的一个后缀之间的 LCP 查询：由于 # 不会出现在 Y中， $S[j..|S|)$ 和 $S[|X|+1+i..|S|)$ 的公共前缀不可能延伸到 # 之后，因此对于所有 $j \in [0..|X|)$ 和$i \in [0..|Y|)$，

$$\mathrm{LCP}\big(X[j..|X|), Y[i..|Y|)\big) = \mathrm{LCE}_S(j, |X|+1+i).$$

哨兵符可能会改变此类一对后缀的相对顺序，但绝不会改变其 LCP 值。将事实 3.1 应用于按后缀数组顺序排列的 S 的后缀后，在 SA_S 中，设 X 的后缀秩集合为 R，那么与$Y[i..|Y|)$ 的 LCP（其中 S 的后缀的秩为 $\rho = \mathrm{iSA}_S[|X|+1+i]$）在 $\max\{r \in R: r < \rho\}$ 或 $\min\{r \in R : r > \rho\}$ 处达到最大值。

前驱与后继查询。给定一个整数集合 K 和一个整数 x， x 在 K中的前驱（分别地，后继）定义为 $\mathrm{pred}_K(x) = \max\{y \in K \mid y \le x\}$（分别地，$\mathrm{succ}_K(x) = \min\{y \in K \mid y \ge x\}$）；若相应集合为空，则前驱或后继未定义。动态前驱数据结构维护集合 K，支持插入和删除操作，并回答前驱与后继查询。我们采用三种经典实现。平衡二叉搜索树（例如 AVL树）支持所有操作，时间复杂度为 O(log $|K|$)，空间复杂度为 O($|K|$)。对于整数全集 $[0..u)$，van Emde Boas（vEB）树 [vEB77] 支持所有操作，时间复杂度为 O(log log u)，空间复杂度为 O(u)，初始化时间为 O(u)。动态融合树 [PT14] 支持所有操作，时间复杂度为 $\mathrm{O}(\log_w |K|) = \mathrm{O}(\log |K| / \log w)$，空间复杂度为 O ($|K|$)；尤其是当 $|K| = \log^{O(1)} n$时，时间复杂度为 O (1)。

不相交集合。并查集数据结构维护全集 $[0..m)$ 的一个划分，该划分初始由单元素集合组成，并支持 find(k) 操作（返回包含 k的集合的代表元）和 union(k, k') 操作（合并分别包含 k 和 k'的集合）。在静态树集合并问题中，预先给定一棵定义在 $[0..m)$ 上的有根树 T；每次合并都将包含非根节点 k 的集合与包含其父节点的集合合并。因此，所有可能的合并在执行任何操作之前便已确定，而操作顺序任意。Gabow 和 Tarjan [GT85] 证明，在这种情况下，任意 q 个操作序列的总时间为 O ($m + q$)。我们采用一种特殊情形： T 为一条路径，其中对于每个 $k \in [0..m-1)$， $k+1$ 是 k的父节点。此时，每个集合都是 $[0..m)$ 中的一个连续区间，每次合并都将两个相邻区间 $[l..k]$ 和 $[k+1..r]$合并。这称为区间并查集问题。

4 线性时间的 LAMAS

我们首先给出一个简单的 O ($n \log d$) 时间 LAMAS 算法，然后将其改进为 O (n) 时间。我们的改进与回答查询的方式有关。

全文中，我们设 $S := X\#Y$（第3节中的字符串），因此 $|S| = |X| + |Y| + 1 \le 2n + 1$。不失一般性，假设 $1 \le d < n$：对于 $d = 0$，有 $\mathrm{l}_i = \mathrm{LCP}(X[i..|X|), Y[i..|Y|))$，并且对于每个 $i < |X|$都有 $\delta_i = 0$（否则为 $\mathrm{l}_i = 0$），即每个位置需要进行一次 LCE查询；对于 $d \ge n$，输出与 $d = n-1$相同，因为每个 $j \in [0..|X|)$ 都满足 $|j - i| \le n - 1$。

4.1 简单算法

主要思路。对于 Y中的每个位置 i，算法都会考虑起始位置与 i 的距离不超过 d 的 X 后缀。这些起始位置构成一个滑动窗口 $[i-d。.i+d]$，并限制在 X 的有效位置内。随着 i 增大，窗口向右移动一个位置。根据事实 3.1，与 $Y[i。.|Y|)$ 共享最长前缀的后缀，在可允许后缀中要么是其字典序前驱，要么是其后继。算法计算这两个候选项的最长公共前缀长度，并将较大者报告为 l_i（平局时任意选择），同时报告所选后缀 $X[j^*。.|X|)$ 的位移 $\delta_i = j^* - i$。因此，在构建 $S = X\#Y$的后缀数组后，LAMAS 可归结为在滑动窗口的后缀数组秩中执行前驱与后继查询。这些秩由平衡搜索树（例如 AVL树）维护，支持在 O (log d) 时间内进行查询和更新。由于窗口总共滑动 O (n) 次，该算法的运行时间为 O ($n \log d$)。

图2 展示了 X = banana、 Y = anana 和 $d = 1$ at $i = 1$ 的这一思路：窗口 {0,1,2} 的秩为 {6,3,9}，查询后缀 $Y[1..|Y|)$ = nana 的秩为 8，其前驱 6（banana）和后继 9（nana）的 LCP 长度分别为 0 和 4；因此， $j^* = 2$ 和 $(l1, \delta_1) = (4, 1)$。

算法。算法 1 给出了简单算法，包含四个步骤：(1) 计算 S 的 SA_S、iSA_S 和最长公共扩展结构（第3节），以支持在 X 和 Y 的后缀之间进行常数时间 LCP 查询（第 1–2 行）。(2) 构造数组 A 和 B，分别存储 X 和 Y的后缀在 SA_S 中的秩（第 3–4 行）。由于以 # 开头的 S 后缀具有最大秩， A 和 B 的值两两不同，且恰好构成集合 $[0..|A| + |B|)$ 。(3) 对每个位置 i，计算 $B[i]$ 在窗口中所有秩里的前驱 r_P 和后继 r_S

$$W_i := \{A[j] : j \in [0..|X|), |j - i| \le d\}.$$

为此，它维护 W_i，其中最多有 $2d + 1$ 个元素，并在窗口滑动时在线更新的 AVL树中进行维护（第5–15行）。（4）它最多使用两次 LCE 查询，每个候选项一次，并报告最大的最长公共前缀长度l_i及相应的位移 $\delta_i = j^* - i$，其中 j^*是 X 中具有最大值的后缀的起始位置（第16–21行）。

正确性源于事实 3.1： W_i 中的秩恰好是可允许后缀的秩$X[j。.|X|)$ 其中 $|j - i| \le d$，并且如第 3节所述，与 $Y[i。.|Y|)$ 的最长公共前缀由 $B[i]$下方或上方最近的秩取得最大值。步骤 1、2 和 4 需要 $O(n)$ 的时间和空间。步骤 3 执行 $O(n)$ AVL 树

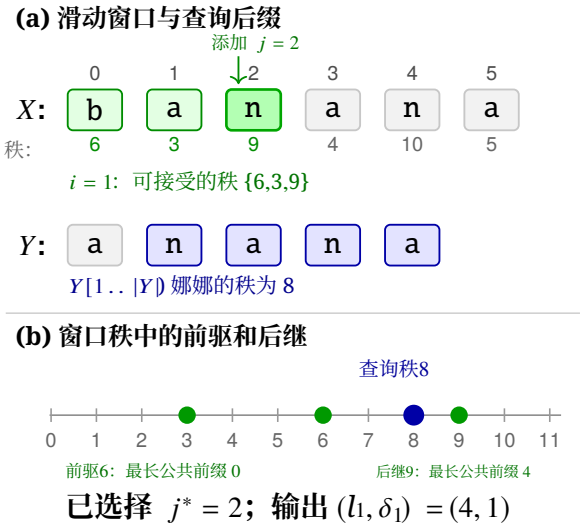


图 2：简单算法的主要思想：(a) 将窗口移动到 $i = 1$ 会添加 X 的位置 2，其秩为 9；(b) 查询秩 8 的前驱和后继的最长公共前缀长度分别为 0 和 4。

算法 1 LAMAS 的简单算法

要求：字符串 X、Y 和位移容差 d确保：数组 l 和 δ1： $S \leftarrow X\#$2：计算 S的 SA_S、iSA_S以及最长公共扩展结构 3：对 $j \in [0$。 。 $|X|)$ 执行 $A[j] \leftarrow \mathrm{iSA}_S[j]$ 4：对 $i \in [0$。 。 $|Y|)$ 执行 $B[i] \leftarrow \mathrm{iSA}_S[|X| + 1 + i]$ 5：初始化一棵空的 AVL树W6：对于 $j = 0$ 到 $\min(d, |X| - 1)$，执行 7： WInsert($A[j]$) 8：对于 $i = 0$ 到 $|Y| - 1$，执行 9：如果 $i > 0$，则 10：如果 $i + d < |X|$，则 11： WInsert($A[i + d]$) 12：如果 $i > d$，则 13： WDelete($A[i - d - 1]$) 14： $r_p \leftarrow W$ Predecessor($B[i]$) 15： $r_s \leftarrow W$Successor($B[i]$) 16： $(\mathrm{l}_i, \delta_i) \leftarrow (0, \perp)$ 17：对于 $r \in \{r_P, r_S\} \setminus \{\perp\}$，执行 18： $j \leftarrow \mathrm{SA}_S[r]$19： $L \leftarrow \mathrm{LCE}(j,$ $|X| + 1 + i)$ 20：如果 $L > \mathrm{l}_i$ 或 $\delta_i = \perp$，则 21： $(\mathrm{l}_i, \delta_i) \leftarrow (L, j - i)$ 22：返回 (l, δ)

在至多包含 $2d + 1$ 个元素的树上进行树操作，即每次耗时 O(log d)，并使用 O(d) 空间。因此，我们得到：

命题 4.1。 可以在 $O(n \log d)$时间内使用 $O(n)$空间解决 *LAMAS* 问题。

例 *4.2*（简单算法）。设 X = 香蕉、 Y = 阿娜娜，且 d = 1。步骤 *1–2* 计算 $S = X\#Y$ 的后缀数组（图 *3a*）以及秩数组 $A = (6, 3, 9, 4, 10, 5)$ 和 $B = (2, 8, 1, 7, 0)$（图 *3b*）。对于 $i = 0$， $W_0 = \{6, 3\}$ 中没有 $B[0] = 2$ 的前驱。其后继秩 *3* 对应于 $X[1$。$.|X|) =$阿娜娜，得到 $(l0, \delta_0) = (5, 1)$。对于 $i = 1$，插入 $A[2] = 9$ 得到

t	0	1	2	3	4	5	6	7	8	9	10	11
$S[t]$	b	a	n	a	n	a	#	a	n	a	n	a
$\mathrm{SA}_S[t]$	11	9	7	1	3	5	0	10	8	2	4	6
逆后缀数组$_S[t]$	6	3	9	4	10	5	11	2	8	1	7	0

(a) 步骤 1： $S = X\#Y$ 及其后缀数组和逆后缀数组。

j (分别为 i)	0	1	2	3	4	5
$A[j]$ = 逆后缀数组$_S[j]$	6	3	9	4	10	5
$B[i]$ = 逆后缀数组$_S[7+i]$	2	8	1	7	0	–

(b) 步骤 2： A 的后缀的秩以及 X 的后缀的秩和 B 的后缀的秩以及 Y

i	0	1	2	3	4
$B[i]$	2	8	1	7	0
W_i	{6, 3}	{6, 3, 9}	{3,9,4}	{9, 4, 10}	{4, 10, 5}
r_p, r_s	$\perp, 3$	6, 9	$\perp, 3$	4, 9	$\perp, 4$
ℓ_p, ℓ_s	$-\infty, 5$	0, 4	$-\infty, 3$	0, 2	$-\infty, 1$
ℓ_i	5	4	3	2	1
j^*	1	2	1	2	3
δ_i	1	1	−1	−1	−1

(c) 步骤 3 和 4：窗口、最近的秩 r_p、 r_s，它们的 LCP 长度 l_p、l_s（其中 $Y[i..|Y|)$），以及输出（$\perp$ 和 $-\infty$表示未定义）。

图 3：在 X = banana、 Y = anana 和 $d = 1$ （示例 4.2）上的简单算法。

$W_1 = \{6, 3, 9\}$（图 2）。 $B[1] = 8$ 的前驱 6 和后继 9 对应的最长公共前缀长度分别为 0 和 4，得到 $(l1, \delta_1) = (4, 1)$。对于 $i = 2$，移除 $A[0] = 6$ 并插入 $A[3] = 4$ 得到 $W_2 = \{3, 9, 4\}$。$B[2] = 1$ 的后继秩为 3，对应于 $j = 1$，得到 $(l2, \delta_2) = (3, -1)$。虽然 $j = 3$ 也达到长度 3，但无需检查其秩 4：任意一个最大化位置都足够。图 3c 展示了完整的执行过程。

4.2 改进1：宇宙缩减

主要思想。第 4.1 节中的算法每次操作需要 O (log d) 的代价，这源于基于比较的前驱搜索。vEB 树可以在大小为 u 的宇宙上，以 O (log log u) 的时间回答前驱查询。然而，我们的秩范围为 $[0..|S|)$，使得 u 过大。我们的关键观察是， d 个连续查询位置的窗口至多涉及 A 中的 $3d$ 个连续条目，以及来自 B的 d 个查询键。由于查询只依赖于键的相对顺序，因此我们可以用局部排名替换键的全局秩，而不会改变查询结果。因此，我们将问题划分为 O (n/d) 个独立块，每个块至多包含 $4d$ 个键，并通过一次全局基数排序，将每个块中的键重新排名为 $[0..4d)$；这样，宇宙规模就足够小，可以使用 vEB 树。随后，可以使用 vEB 树在 O (log log d) 的时间内处理每个滑动窗口，从而使每个块的处理时间为 O (d log log d)，总体时间为 O (n log log d)。

一个辅助问题。我们通过定义并求解问题2，将寻找每个查询后缀秩在其可允许窗口内的前驱和后继这一任务单独提取出来。

问题 2 (滑动窗口前驱/后继 (SWPS))。 给定正整数 N 、数组$a = (a_0, \ldots, a_{3N-1})$和 $b = (b_0, \ldots, b_{N-1})$，其中元素是 $[0..4N)$ 中的两两不同整数，[2] 对于每个$i \in [0..N)$，输出窗口 $\{a_j : j \in [i..i + 2N]\}$中 b_i的前驱和后继；如果不存在这样的元素，则将答案报告为未定义。

[2]某些条目可能未定义，而 b 的条目数可能少于 N，以处理 X和 Y的边界；未定义的 ed 条目将被忽略略。

P命题 4.3. *SWPS* 问题可以在 $O(N \log\log N)$ 时间内求解，使用 $O(N)$ 空间 $e.$

证明。 在 $[0..4N)$ 上初始化一棵 vEB 树；将 $a_0, \ldots, a_{2N}$ 中已定义的值插入其中。这需要 O (N) 的时间和空间。对于 $i \in [0..N)$，我们在树中查询 b_i的前驱和后继，然后删除 a_i 并插入 a_{i+2N+1} （如果 $i + 2N + 1 < 3N$；跳过未定义的条目），使树恰好包含 b_{i+1}的窗口。每次操作需要 O $(\log\log N)$ 时间，共有 O (N) 次操作。□

算法。 简单算法的步骤 1、2 和 4 保持不变；我们将步骤 3 替换为归约到 O(n/d) 个 SWPS 实例。

归约的第一步是将 B 划分为块 $B_k := B[kd..(k+1)d)$，每块包含 d 个位置，其中 $k \in [0..\lceil |Y|/d \rceil)$，并将每个 B_k 与 $A_k := A[(k-1)d..(k+2)d)$ 块配对，该块包含 $3d$ 个 A条目。这恰好构成 SWPS 结构：对于 $i' \in [0..d)$，$B[kd+i']$ 的窗口为 $A[(k-1)d+i'..(k-1)d+i'+2d]$，即 A_k 的位置 $[i'..i'+2d]$。$[0..|X|)$ 之外的 A_k 条目未定义，而 B 的最后一个块可能短于 d。A 的连续块彼此重叠 $2d$ 个位置，因此 A 的每个条目最多属于三个块，所有实例的总规模为 O (n)。

第二步是将每个实例的值域缩减到 $[0..4d)$，以便应用命题 4.3。对于每个 k，我们构造集合 M_k，其中包含 A_k和 B_k的（已定义）值，每个值都标记了其所属的块 k，以及它在 A_k 或 B_k中的索引。我们通过一次基数排序将所有 M_k一起排序：先在值域 $[0..|S|)$上，按值对所有标签进行计数排序，再按块索引进行稳定计数排序。此过程耗时 O (n)。将每个值替换为其在排序后的 M_k 中的位置，即可得到值域为 $[0..4d)$ 的数组 A'_k和 B'_k，它们与 A_k 和 B_k具有相同的相对顺序；因此，前驱和后继关系得以保留。

最后，我们通过命题 $a := A'_k$，$b := B'_k$，$N := d$，在 4.3中，以 $O(d \log\log d)$时间求解每个 SWPS 实例，并为所有实例重复使用（并在 $O(d)$ 时间内重新初始化）同一棵 vEB 树。

复杂度。步骤 1、2 和 4 以及基数排序的总时间为 O (n)，SWPS 实例的时间为 O $(n/d) \cdot$ O $(d \log\log d)$ = O $(n \log\log d)$。如果 $d = \log^{O(1)} n$，我们将 vEB 树替换为动态融合树 [PT14], , 它支持对至多 $2d + 1$ 个键执行的每种操作，并可在 O $(\log_w d)$ = O$(\log d/\log w) \subseteq$ O $(\log d/\log\log n)$ = O (1) 时间内完成，其中 $w = \Omega(\log n)$ 是以位为单位的字长。

命题 4.4。 *LAMAS* 问题可以在$O(n \log\log d)$ 时间内求解，所需$O(n)$ 空间。如果$d = \log^{O(1)} n$，则可以在$O(n)$时间内求解，所需$O(n)$空间。

4.3 改进2：窗口分解与并查集

主要思路。我们消除了对 d 的依赖，证明每个 SWPS 实例都能在 O(N) 时间内求解，因此 LAMAS 能在最优的 O(n) 时间内求解。我们注意到，SWPS 的窗口具有严格的结构。将输入数组 a 划分为三个等长部分：（左侧） a_L、（中间） a_M和（右侧） a_R。对于 b中的每个查询键 b_i，其窗口由 a_L的一个后缀、完整的 a_M以及 a_R的一个前缀组成，如图 4 所示。中间部分 a_M 是静态的，因此一次有序扫描即可回答关于它的所有查询。随着 i 增大，a_L 的后缀只会缩短；若按 i递减的顺序处理查询， a_R 的前缀也只会缩短。因此，其余所有查询都是在一个仅经历删除操作的有序序列上进行的前驱/后继查询，这正是可在线性时间内求解的区间并查集问题。

由于 $i \in [0..N)$， b_i 的每个窗口都可分解为

$$[i..i+2N] = [i..N) \cup [N..2N) \cup [2N..2N+i].$$

也就是说，令 $a_L := a_0, \ldots, a_{N-1}$、 $a_M := a_N, \ldots, a_{2N-1}$和 $a_R := a_{2N}, \ldots, a_{3N-1}$。分别用 $\mathrm{pred}_L(i)$、$\mathrm{pred}_M(i)$ 和 $\mathrm{pred}_R(i)$ 表示 b_i 在 $\{a_i, \ldots, a_{N-1}\}$、$\{a_N, \ldots, a_{2N-1}\}$ 和 $\{a_{2N}, \ldots, a_{2N+i}\}$ 中的前驱（succ 同理），则有

$$\mathrm{pred}(i) = \max\{\mathrm{pred}_L(i), \mathrm{pred}_M(i), \mathrm{pred}_R(i)\},$$

$$\mathrm{succ}(i) = \min\{\mathrm{succ}_L(i), \mathrm{succ}_M(i), \mathrm{succ}_R(i)\},$$

其中 u 未定义的值将被忽略。我们使用以下工具来回答这三类查询。

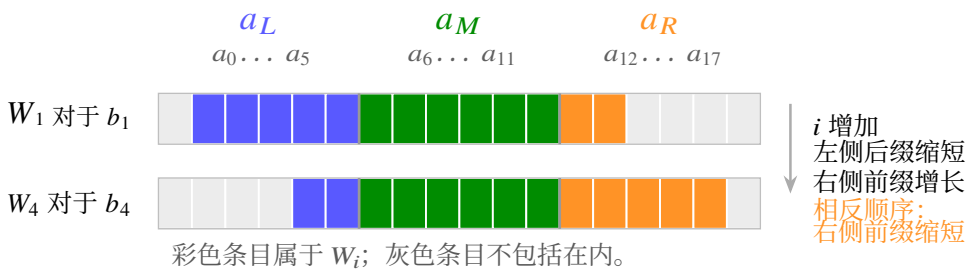


图 4： $N = 6$的 SWPS 窗口分解。

引理 4.5 (区间并查集 [GT85])。 设 $e_0, \ldots, e_{m-1}$ 是一个元素序列，初始时所有元素均存在。有一种数据结构支持以下操作：删除(k)，将 e_k 标记为已删除；后继(k)，返回满足 $e_{k'}$ 存在的最小 $k' \geq k$ （如果不存在这样的 k'，则报告不存在）；以及前驱(k)，返回满足 $e_{k'}$ 存在的最大 $k' \leq k$ （如果不存在这样的 k'，则报告不存在）。任意 q 次此类操作总共耗时 $O(m + q)$。

证明。我们将 $[0。。m)$ 划分为以下形式的区间：每个区间 $[l。。r]$ 均满足： e_l、。。。、e_{r-1} 已被删除，而 e_r 仍然存在；但最右侧区间除外，该区间中的所有元素都可能已被删除。初始时，每个元素都构成一个单元素区间。每个区间都作为一个集合存储在并查集数据结构中，其代表元存储 l 和 r。操作 delete(k) 将包含 k 的区间与包含 $k+1$ 的区间合并（如果 $k < m-1$）；这会保持不变式，并且除并操作本身外只需 O(1) 时间。对于 next(k)，我们找到包含 k 的区间 $[l。。r]$，如果 e_r 存在，则返回 r（否则不存在这样的元素）。对于 prev(k)，我们找到包含 k 的区间 $[l。。r]$；如果 $k = r$ 且 e_r 存在，则返回 k；否则， e_k 已被删除，我们返回 $l-1$ （如果 $l > 0$；否则不存在这样的元素），该元素根据这些区间的极大性必然存在。由于每次合并都会将包含某个 k 的集合与包含 $k+1$ 的集合合并，因此这是区间并查集问题：并查集树是一条满足 parent$(k) = k+1$ 的路径，该路径预先固定，而删除操作可以按任意顺序到达。任意 q 次操作总共耗时 O$(m+q)$ [GT85]。□

引理 4.6。 *SWPS* 问题可以在$O(N)$ 时间内解决，使用$O(N)$ 空间。

证明。 由于 a 和 b 的所有（已定义）值都是 $[0..4N)$ 中两两不同的整数，我们首先使用计数排序，在 O(N) 时间内获得这些值的有序序列，并为每个值标注其来源（a_L、 a_M、 a_R 或 b）以及它在 a 或 b 中的索引。对该序列进行一次扫描即可得到：

- a_L、 a_M 和 a_R 的（已定义）值的有序序列 $c_0^L < c_1^L < \cdots$、$c_0^M < c_1^M < \cdots$ 和 $c_0^R < c_1^R < \cdots$，以及每个已定义的 a_j 在其所属部分的有序序列中的秩 $\rho(j)$；
- 对于每个 $i \in [0..N)$，分别小于 b_i 的 a_L、 a_M 和 a_R 的值的数量 π_i^L、 π_i^M 和 π_i^R。

注意，在 P 的任意子集 a_R 中， b_i 的前驱（相应地，后继）是 P 中在 c^R 里秩不大于 $\pi_i^R - 1$（相应地，不小于 π_i^R）的元素中秩最大的（相应地，最小的）元素；对于 a_L 和 a_M，同理。

中间部分。 由于集合 $\{a_N, \ldots, a_{2N-1}\}$ 对所有查询都相同，因此对于每个查询，我们有 $\text{pred}_M(i) = c^M_{\pi_i^M - 1}$ （如果 $\pi_i^M = 0$ 则未定义），以及 $\text{succ}_M(i) = c^M_{\pi_i^M}$ （如果 π_i^M 等于 a_M 中已定义值的数量，则未定义）。每次查询耗时 O(1)。

右侧部分。 我们按照 $i = N-1$、 $N-2$、.....、0 的顺序处理查询，使得集合 $\{a_{2N}$、.....、$a_{2N+i}\}$ 仅会缩小。我们在有序序列 c^R 上初始化引理4.5 中的数据结构（因此， m 是 a_R中已定义值的数量）。对于从 $i = N-1$ 递减至 0 的每个值，我们首先回答关于 b_i的查询：$\text{pred}_R(i) = c^R{}_{\text{prev}(\pi_i^R - 1)}$（如果 $\pi_i^R = 0$未定义，或prev未报告任何元素，则未定义），以及 $\text{succ}_R(i) = c^R{}_{\text{next}(\pi_i^R)}$（如果 $\pi_i^R = m$未定义，或next未报告任何元素，则未定义）；然后，如果 a_{2N+i} 已定义，我们执行delete$(\rho(2N+i))$; 根据归纳法，当

关于 b_i 的查询被回答时， a_{2N}、……、 a_{2N+i} 中恰好所有已定义的值都存在，正如要求的那样。我们总共执行$O(N)$次操作，根据引理 $O(N)$，总耗时为 4.5。

左半部分。这是对称的：我们按 $i = 0$、1、……、$N-1$ 的顺序处理查询，因此集合 $\{a_i$、……、$a_{N-1}\}$ 只会缩小。我们在 c^L 上初始化引理 4.5 的数据结构，对于 $i = 0$ 到$N-1$，我们使用 prev($\pi_i^L - 1$) 和 next(π_i^L)，完全按照上文的方式回答 b_i 的查询，然后在 a_i 有定义时执行delete($\rho(i)$)。同样，这需要 O (N) 时间。

我们将部分答案合并，对于每个 i，在 O (1) 时间内分别得到 pred(i) 和 succ(i)，从而获得 O (N) 的时间和空间复杂度。 □

最终算法。该算法与第 4.2 节中的算法相同，只是以引理 4.6 替代命题 4.3：步骤 1、2、4 和全集缩减需要 O(n) 时间，而 O(n/d) 个 SWPS 实例总共需要 O(n/d) · O(d) = O(n) 时间。LCE 数据结构和排序数组需要 O(n) 空间，此外，引理 4.5 的数据结构需要 O(d) = O(n) 空间，这些数据结构会在各个实例间重复使用。

The定理 4.7。 *LAMAS* 问题可在 $O(n)$时间内求解，使用 $O(n)$空间，适用于任意 $d \geq 0$.

该界是 最优的： Ω(n) 读取输入需要的时间为该时间复杂度，且这与计算 MS 数组所需的时间相匹配 [CL90]。

5 实验评估

我们的实验评估围绕五个问题展开。

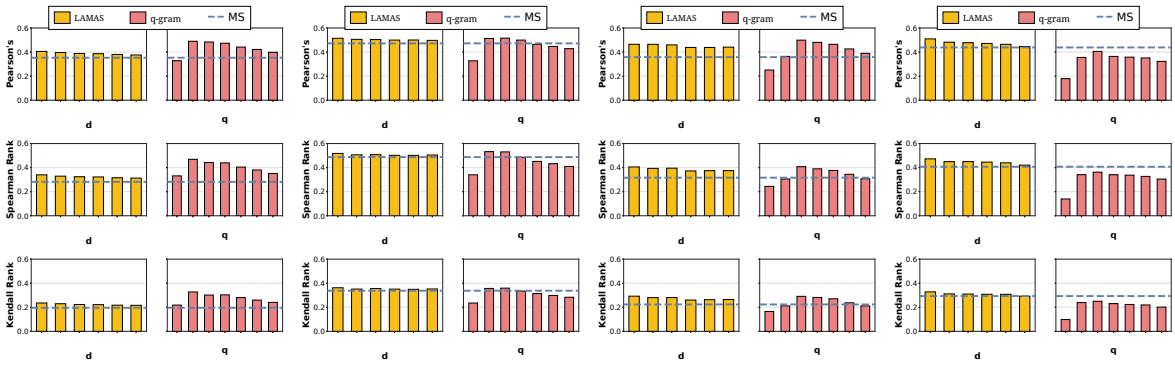
Q1 随着序列 LAMAS 与 ED 的相关性会如何变化？随着序列 分歧度 增大？（第 5.1节）。*Q2* LAMAS 对时间序列对的分类效果如何？（第 5.2节）。Q3 *LAMAS* 在 ED 下恢复前 LAMAS 个结果的准确度如何？（第 K 5.3节）。Q4 *LAMAS* 通过聚类恢复参考类别的效果如何？（第 5.4节）。Q5 我们的 $O(n)$时间 LAMAS 算法在运行时间和空间方面的效率如何？（第 5.5节）。

我们使用 20 个真实数据集以及合成数据集。我们将 O(n) 时间的 LAMAS 算法与以下方法进行比较：(1) 我们自行实现的 MS；(2) 我们自行实现的 q-gram 距离；(3) Edlib [SS17]，的 ED 实现，它采用高效的 Myers 位向量算法；(4) 第 4.1 节中 O($n \log d$) 时间的算法，称为 LAMAS（简单版）；(5) 第 4.2 节中 O($n \log \log d$) 时间的算法，称为 LAMAS (vEB)。所有 LAMAS 实现和 MS 均使用 SDSL-lite [GBMP14]。对于分类实验，我们还与带状动态时间规整 [SC78], 进行比较；该方法限制对齐位置之间的位移，因此是一个自然的基线。对于前 K 项检索实验，我们还与 MinSearch [ZZ20a] 的一个改编版本进行比较。由于编辑距离嵌入方法 [CGK16]，在我们的设置中始终逊于 LAMAS，因此我们不报告其结果。例如，在第 5.1 节的实验中，它与 ED 的平均皮尔逊相关系数比表现最差的竞争方法低 70%。

所有实验均在一台主频为 2.50GHz、配备 188GB 内存的 Intel(R) Xeon(R) Gold 6248CPU 上运行。我们的代码和数据集可在以下网址获取：`https://github.com/lorrainea/LAMAS`。

5.1 与编辑距离的相关性

任务与数据。我们评估 LAMAS、MS和 q-gram 距离与编辑距离（ED）的相关程度，以及它们与编辑距离的相关性如何随分歧度变化（即对序列执行的编辑操作数量除以其长度）。对于LAMAS和MS，我们将匹配长度转换为基于对称 ACS 的不相似度；参见 [UBTC06]。我们使用 400 对长度为 $n \approx 100{,}000$ 的 DNA 序列，这些序列由 Wavefront 比对算法的生成数据集工具生成 [MMME21], ，该工具可模拟基因组变异：在分歧度为 15%、20%、25% 和 30% 的每个水平上生成 100 对序列。



(a) (b) (c) (d)

图 5：对于 100 对长度为 $n \approx 100$，000、分歧度分别为：(a) 15%、(b) 20%、(c) 25% 和 (d) 30% 的序列，LAMAS 与编辑距离、MS 与编辑距离以及 q-gram 距离与编辑距离之间的皮尔逊相关系数、斯皮尔曼秩相关系数和肯德尔秩相关系数。

度量。 我们计算LAMAS 在不同 $d \in${500、1000、.....、3000}下的值、 q-gram 距离在不同 $q \in${8、9、.....、14}下的值，以及MS。我们使用三种广为人知的相关性度量评估它们与编辑距离的相关性：皮尔逊 [Pea95],斯皮尔曼 [Spe04], 和肯德尔秩 [Ken38] 相关性。

结果。 图 5 显示 LAMAS 与编辑距离的相关性显著强于 MS；平均高出 12.4%，最高可达 30.8%。 LAMAS 的表现也优于 q-gram 距离：在所有测试参数值和分歧度水平下取平均，其皮尔逊相关系数、斯皮尔曼秩相关系数和肯德尔秩相关系数分别高出 11.3%、10.6% 和 12.2%。在分歧度为 30% 时，其优势进一步扩大，提升幅度分别最高达 42.4%、45.1% 和 48.3%。LAMAS甚至优于 q-gram 距离在该分歧度水平下的最佳配置 26%–31%。

5.2 配对分类

任务和数据。 我们评估共享子串的位置是否有助于区分同一类别中的时间序列与属于不同类别的时间序列。我们使用表 1 中的 15 个真实数据集，每个数据集包含 N 条带标签时间序列。我们将每个数据集中的每条时间序列离散化为长度为 n 的字符串，具体如下文所述。我们构造字符串对；如果源时间序列具有相同的类别标签，则称该配对为 正样本对 ，而负样本对则表示其他情况。例如，在 PenDigits [UCRm] 数据集中，同一数字的两次记录构成一个正样本对。然后，我们使用下文所述的某一种测试度量计算该配对的相似度分数，并将其与验证集选定阈值进行比较，以预测该配对是否为正样本对。类别标签用于建立真实值，以进行参数选择和评估。

从时间序列到字符串。我们使用标准的 SAX 预处理 [LKWL07] 将每个数值维度转换为字符串： z归一化、重叠加权分段聚合近似（PAA） [KCPM01], 以及使用高斯断点进行离散化。如果数据集中的所有时间序列长度均为L，则将字符串长度设为 $n = \min\{128, L\}$。否则，我们使用 $n = 128$；重叠加权聚合会将长度小于 128 的序列映射为包含 128 个符号的字符串。对于每个数据集，我们通过最大化验证集上的 F_1 [MRS08]，为每种方法选择字母表大小 $\sigma \in \{2, 4, 6, 8, 12, 16\}$、参数和决策阈值。这样，每种方法都可以使用其验证集选定的最佳配置。我们使用所选设置报告测试配对上的 F_1 。

从匹配结果到配对判定。 我们使用五种统计量来汇总匹配长度。对于每个$k \in${1、2、3、5、8}，我们将 $f_k(X, Y)$ 定义为以下两项的平均值：在 X 中， Y里匹配长度至少为 k 的位置所占比例，以及交换 X 和 Y 后对应的比例。对于每种统计量，我们使用根据拟合对估算的均值 μ_k和标准差 s_k对 $f_k(X, Y)$ 进行标准化，并

表 1：数据集和正类测试 F_1。 n 是 SAX 长度， N 是序列总数。最佳（次佳）结果以粗体（下划线）标示。

			LAMAS			MS		q-gram			ED		带状动态时间规整
数据集	n	N	F_1	σ	d	F_1	σ	F_1	σ	q	F_1	σ	F_1
EOGHorizontalSignal [UCRg]	128	724	**0.7383**	6	19	0.6958	8	0.6956	6	4	0.7362	6	0.7102
唐人街 [UCRe]	24	363	**0.7612**	16	0	0.6276	8	0.7074	6	3	0.7111	8	0.6851
WordSynonyms [UCRq]	128	905	**0.6894**	6	37	0.6748	4	0.6772	6	8	0.6821	4	0.6797
肉类 [UCRk]	128	120	0.7843	12	1	0.7442	8	**0.7880**	12	8	0.7840	12	0.7789
ACSF1 [UCRb]	128	200	0.7348	4	20	0.7244	4	**0.7521**	2	6	0.7016	4	0.6870
CharacterTrajectories [UCRd]	128	2858	0.8840	6	10	0.7809	16	0.7905	4	8	0.8726	2	**0.8991**
ArticularyWordRecognition [UCRc]	128	575	0.8852	8	12	0.8134	16	0.8415	16	2	0.9218	8	**0.9220**
GesturePebbleZ1 [UCRj]	128	304	0.8609	4	9	0.7248	12	0.7540	4	1	**0.8951**	4	0.7699
UWave 手势库 [UCRp]	128	440	0.7718	4	16	0.6927	2	0.6327	4	4	**0.8169**	2	0.8050
PenDigits [UCRm]	8	10992	0.7524	2	1	0.7145	2	0.7053	2	4	0.7440	4	**0.7999**
FacesUCR [UCRh]	128	2250	0.6892	6	5	0.6378	6	0.6688	4	3	**0.7383**	6	0.7294
ShakeGestureWiimoteZ [UCRn]	128	100	0.8101	6	21	0.6748	8	0.5484	12	1	0.8210	8	**0.8407**
Fish [UCRi]	128	350	0.6846	16	6	0.6670	16	0.6900	8	1	**0.7507**	8	0.6786
AconityMINIPrinterSmall [UCRa]	128	881	0.7012	8	0	0.6660	8	0.6651	4	1	0.6840	8	**0.7195**
MixedShapesSmallTrain [UCRl]	128	2525	0.7184	12	13	0.6569	16	0.6299	16	8	**0.7203**	12	0.7096
均值 ±标准差	_	_	0.7644 ±0。0688	_	_	0.6997 ±0.0516	_	0.7031 ±0。0741	_	_	**0.7720** ±0。0770	_	0.7610 ±0。0792

应用双曲正切(($f_k(X,\ Y) - \mu_k)/(2s_k)$)，将其贡献限制在(−1,)范围内。基于双曲正切的归一化已用于组合匹配分数 [JNR05]。LAMAS 和 MS 使用相同的评分流程。

我们使用验证集选定的阈值对每个配对进行分类。对于多元时间序列，在各维度之间对分数取平均。ED 和 q-gram 距离会转换为相似度分数。作为基线方法，我们在 SAX 之前对归一化的 PAA 序列应用带状动态时间规整 [RK04]，并在验证配对上选择其规整带宽和决策阈值。

参数选择与评估。 我们保留每个数据集的训练/测试划分，并将训练字符串划分为拟合（60%）和验证（40%）子集，同时保持类别比例。在每个子集中，我们构造均衡的正/负样本对，最多分别使用 2000 个拟合对、2000 个验证对和 20000 个测试配对。拟合对用于确定特征标准化；验证配对用于选择每种方法的字母表大小和阈值，以及 LAMAS 的 $d \in \{0, \ldots, n-1\}$ 以及基线方法的 $q \in \{1, 2, 3, 4, 6, 8\}$。带状动态时间规整的半径和阈值通过最大化验证集上的 F_1来选择。所有配置使用相同的子集和配对，且不同子集之间不共享序列。我们使用 F_1 [MRS08]在测试配对上评估选定的配置。对所有配对均预测为正样本，在我们的均衡配对上可获得 2/3 的得分。

结果。在表 1 的三个编辑距离代理指标中，LAMAS 的平均值最高，为 F_1 (0.7644)，分别比 MS 和 q-gram 距离高出 9.2% 和 8.7%，接近编辑距离（ ED ）的有效性（ 0.7720 ）。带状动态时间规整的平均值较低，为 F_1 (0.7610)。在全部 15 个数据集上，LAMAS 均优于 MS；在所有方法中，LAMAS 在 3 个数据集上排名第一，在 7 个数据集上排名第二。对这 15 个数据集进行的双侧配对 Wilcoxon 符号秩检验证实，LAMAS 显著优于 MS（$p < 0.001$）和 q-gram 距离（$p < 0.01$）。与编辑距离（ED）和带状动态时间规整相比，差异在统计上并不显著（分别为 $p = 0.64$ 和 $p = 0.98$）。所选的 d远小于 n （在 MS 中为 $d = n - 1$），这表明在分类中限制匹配位置具有优势。

5.3 Top-K检索

任务和数据。 我们评估以下方法的有效性：LAMAS、MS、 q-gram 距离，以及 MinSearch [ZZ20a]在检索与 ED 相同的、与查询字符串（模式）最相似的前K 个字符串方面的表现。我们纳入了 MinSearch，

表 2：直接检索 F_1@10。LAMAS 和 MS 使用相同的五统计量评分。 n 表示 SAX 长度， N 表示序列总数。最佳（次佳）分数以粗体（下划线）标出。

数据集	n	N	LAMAS	d	MS	q-gram	q	MinSearch
EOGHorizontalSignal [UCRg]	128	724	0.8500	5	0.3133	0.4467	8	**0.9167**
唐人街 [UCRe]	24	363	**0.9033**	1	0.8200	0.8233	3	0.3740
WordSynonyms [UCRq]	128	905	0.6467	10	0.2167	0.2867	4	**0.6567**
肉类 [UCRk]	128	120	**1.0000**	0	0.9467	0.9533	4	0.9900
ACSF1 [UCRb]	128	200	**0.9367**	1	0.5567	0.5500	4	0.9200
CharacterTrajectories [UCRd]	128	2858	0.5467	5	0.1333	0.2667	4	**0.8867**
ArticularyWordRecognition [UCRc]	128	575	**0.6567**	5	0.2033	0.1967	8	0.5333
GesturePebbleZ1 [UCRj]	128	304	**0.7600**	8	0.4467	0.4600	3	0.5104
UWave 手势库 [UCRp]	128	440	**0.7400**	15	0.3033	0.3967	4	0.6861
PenDigits [UCRm]	8	10992	**0.9800**	0	0.8267	0.8267	3	0.6606
FacesUCR [UCRh]	128	2250	**0.6033**	6	0.2567	0.3000	4	0.2595
ShakeGestureWiimoteZ [UCRn]	128	100	**0.7300**	5	0.4500	0.4900	4	0.3213
Fish [UCRi]	128	350	0.5867	3	0.3567	0.4867	8	**0.9767**
AconityMINIPrinterSmall [UCRa]	128	881	**0.6000**	7	0.1233	0.2300	3	0.4370
MixedShapesSmallTrain [UCRl]	128	2525	**0.7333**	10	0.4933	0.4667	4	0.6733
均值 ±标准差	—	—	**0.7516** ±0。1502	—	0.4298 ±0.2593	0.4787 ±0.2289	—	0.6535 ±0。2448

一种先进的前K 检索方法，可在 ED 下提供概率正确性保证。[3] 我们不与较新的启发式方法 [LPS23] 进行比较，该方法的优势在于效率，而我们的重点是有效性。

我们使用表2中的全部15个数据集。对于包含多变量时间序列的数据集，我们从每条时间序列中均匀随机保留一个维度。对于每个数据集，我们使用从其测试划分中采样的10个验证查询和30个互不相交的测试查询，搜索其原始训练序列。所有方法使用相同的序列和查询。序列经过 z归一化，并使用分段聚合近似和高斯断点转换为SAX字符串 [LKWL07]。所有方法均使用 $\sigma = 8$。

评分与参数选择。对于 MS 和 LAMAS，我们使用第5.2节中描述的相同五统计量相似度评分，并基于无标签数据库对进行归一化。 q-gram 基线根据子串计数向量之间的 l_1 距离进行排序。我们保留 LAMAS 的位移容差 $d \in \{0, \ldots, n-1\}$，以及先前通过最大化验证 F_1@10 选择的 q-gram 长度 $q \in \{1, 2, 3, 4, 6, 8\}$；这些选择在所有 K 中保持不变。

评估与结果。编辑距离提供真实值。令 r 为第 K小的距离， L 为距离小于 r的记录， E 为距离等于 r的记录， R 为返回的记录。 $F_1@K = 2c/(|R| + K)$，其中$c = |R \cap L| + \min\{|R \cap E|, K - |L|\}$。因此，边界距离相同的记录可以互换，而漏掉距离更近的记录会受到惩罚。我们先在每个数据集内对测试查询的 $F_1@K$ 取平均值，再对各数据集等权求平均。

表 2 显示，在 $K = 10$ 下，LAMAS 在全部 15 个数据集上均优于 MS 和 q-gram 距离（跨数据集的 Wilcoxon 符号秩检验，两者均为 $p < 0.001$）。相对于 MS 的提升为 74.9%。LAMAS 在 15 个数据集中的 11 个上也优于 MinSearch，但差异不具有统计显著性（$p = 0.107$）。图 6 报告了对 K 的敏感性。在每一个评估的 K 上，LAMAS 均优于 MS 和 q-gram 距离，且跨数据集的变异更小。对于 $K \geq 5$，它平均也优于 MinSearch（例如，其 F_1@100 为 0.8565，而对比值为 0.5656）。d 的值小于 n，表明与 MS 相比，LAMAS 限制匹配位置的有效性（$d = n - 1$）。

[3]我们改造了作者的代码 [ZZ20b]，使其能够处理短字符串，并在未填充的输入上安全地验证编辑距离。不添加穷举回退，因此返回的结果可能少于 K。我们将内部 gram 长度设为 $q_{\text{MSrch}} = \lceil \log_8 n \rceil$。我们针对每个 $\alpha \in \{0。5，1、2、3\}$分别选择控制所搜索划分层级的参数，方法是最大化验证集上的平均 $K@F_1$。K。

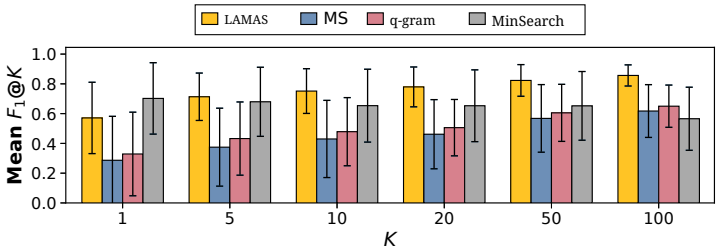


图 6： F_1@K 与 K的均值；误差线表示样本标准差；大小为 $\leq K$ 的数据集不包括在内。

5.4 聚类

任务。 我们使用调整兰德指数评估聚类质量 [HA85], 归一化互信息 [SG02], 纯度 [MRS08], 以及类别支持加权的 F_1 [sci], 在使用一对一分配将聚类最优匹配到参考类别后计算；数值越高越好，调整兰德指数取值范围为 [−0。5, 1] 且所有其他度量的取值范围为 [0, 1]。

我们使用真实数据集，以及为在以下条件下实现清晰的类别分离度而选择的较小真实数据子集 ED.

表 3：聚类数据集和选定参数： N 长度为 n 的字符串，字母表大小为 σ，以及 k 个类别。参数 d 和 q 在前两个数据集的训练数据上选择，在其余三个数据集的评估子集上选择。

数据集	N	n	σ	k	d	q
Crop [UCRf]	16,800	128	8	24	8	6
星光曲线 [UCRo]	8,236	128	8	3	16	8
CinCECGTorso [DBK+19]	235	512	8	4	24	4
心电图5000 [DBK+19]	408	140	8	5	26	2
ADFA-LD [CH13]	24	500	≈ 170	2	3	6

真实数据集。我们表明，与 MS 和 q-gram 距离相比，LAMAS 能够更有效地对真实数据集进行聚类，同时仍可与编辑距离竞争。我们使用两个真实数据集（见表 3）：Crop，包含长度为 $L = 46$ 的图像衍生时间序列；以及星光曲线，包含长度为 $L = 1024$ 的恒星亮度序列。我们对它们的测试划分进行聚类，并使用训练数据选择参数。具体而言，我们采用完全链接的凝聚式聚类 [JD88], , 将每棵树状图切分为k 个聚类，其中 k 为参考类别的数量（表 3）。对于 LAMAS 和 MS，不相似度为 1 –l/n，其中 l 是双向上的平均匹配长度； q-gram 距离是子串计数向量（包括重叠部分）的 l_1 差异。

两个数据集均采用标准差为 max(1，0。01L) 的高斯平滑、 z-归一化、重叠加权分段聚合近似 PAA [KCPM01], 以及使用标准正态断点的SAX 离散化 [LKWL07]。我们使用 $n = 128$ 和 $\sigma = 8$；对于 Crop，这会在不增加信息的情况下增加表示长度。每个数据集固定抽取 500 个训练序列作为均匀样本，抽取时不查看标签或距离，并通过在 $d \in \{0，1、2、4、8、16、32、64\}$ 和 $q \in \{1，2、3、4、6、8\}$ 上最大化调整兰德指数（ARI）来选择参数；出现并列时，优先选择较小的值。

LAMAS 在全部四项度量上均优于 MS 和 q-grams（图 7）。对于 Crop，其调整兰德指数比 MS 高 146%，比 q-grams 高 154%，而 ED 表现略好。对于 星光曲线，LAMAS 在全部四项度量中均排名第一；其调整兰德指数和加权 F_1 较 MS 分别提升 6.8% 和 12.8%，较 q-grams 分别提升 13.1% 和 4.5%；其调整兰德指数也比 ED 高 96%。所选边界 $d = 8$,16 远小于 $n = 128$，而 MS 对应于 $d = 127$。

由 ED 选出的案例研究。我们表明，与 MS 和 q-gram 距离相比，LAMAS 能够更好地恢复参考类别结构，表现往往甚至优于 ED 本身，即便是在特意选择来有利于类别分离的集合上也是如此

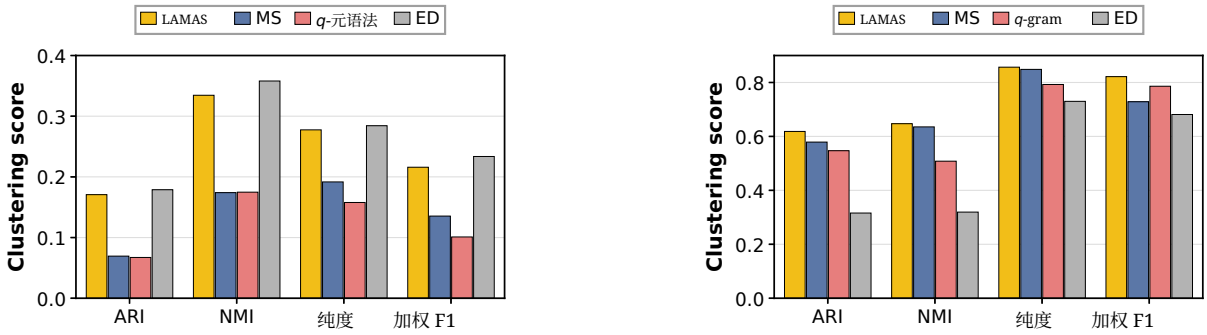


图7：在完整测试划分上进行聚类的效果：(a) Crop 和 (b) 星光曲线。

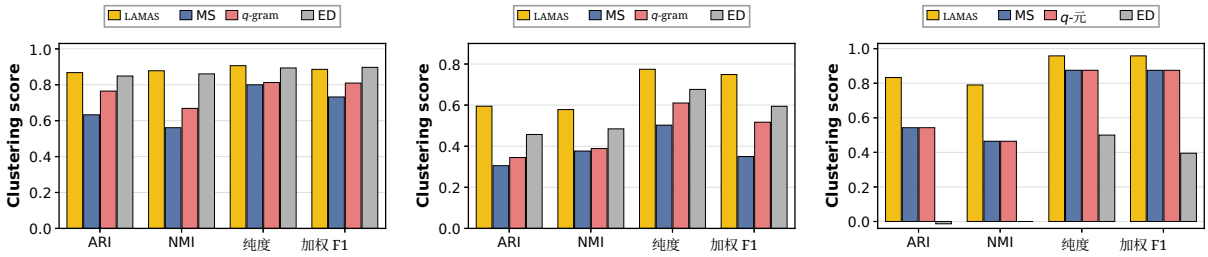


图8：(a) CinCECGTorso、(b) ECG5000和 (c) ADFA-LD的聚类结果。

（这种分离是基于 ED 的。我们使用三个真实数据集（见表 3）：CinCECGTorso、ECG5000以及系统调用数据集 ADFA-LD。对于 CinCECGTorso 和ECG5000，我们采用与上文相同的归一化、分段聚合近似（PAA）和 SAX 离散化方法，但不进行高斯平滑。对于 ADFA-LD，我们合并连续重复的调用，使用唯一的边界标记将每个类别内的轨迹串接起来，并以 250 为步长提取长度为 500 的窗口；系统调用即为字母表中的字母。构建这些字符串后，我们根据 ED 计算每个类别的类别中心对象，并保留那些到最近其他类别中心对象的距离与到本类别中心对象的距离之差最大的字符串。所得子集大小见表 3。参考标签用于指导子集选择、在选定子集上调整 d 和 q 并进行评估；它们不会作为成对字符串比较分数或聚类合并决策的输入。

如图8所示，尽管这些子集经过选择，旨在提高编辑距离下的类别分离度，LAMAS 的表现仍优于 MS 和 q-gram 距离，并且在几乎所有比较中都优于编辑距离。例如，在ECG5000上，LAMAS 的调整兰德指数和加权 F_1 分别比编辑距离高约 31% 和 26%；在ADFA-LD上，其调整兰德指数和归一化互信息分别比最强竞争方法高 53% 和 70%。

5.5 时间和空间效率

真实DNA测序数据。我们使用牛津纳米孔技术公司（ONT）的两个读段数据集（测序DNA片段）：ONT1，读段长度为 40,000–50,000，分歧度为 12% [MEG+23]；以及 ONT2，包含 50 对长度为 500,135–848,888 的超长读段，这些读段已比对至 CHM13 v1.1 人类参考基因组，平均分歧度为 6.1% [Pai24]。对于这两个数据集，我们为每个字符串对（X, Y）设置 $q = 4$ and$d = \lfloor 0.1 \cdot \min(|X|, |Y|) \rfloor$。我们将 LAMAS 与其未优化版本以及基于字符串的距离度量进行比较。

图9 显示，在ONT1 读段上，LAMAS 比编辑距离慢 11%，但在长度长得多的ONT2 读段上，速度快 8.5 倍。在ONT1上，LAMAS 分别比 LAMAS（简单版）和 LAMAS (vEB) 快 61% 和 65%；在ONT2上则分别快 78% 和 80%。其运行时间与ONT1上的 q-gram 距离相当，但在ONT2上更长。在ONT1和ONT2上，LAMAS 分别比 MS 慢 27% 和 26%；为施加位移约束而增加的这点开销并不大，却能提升有效性，

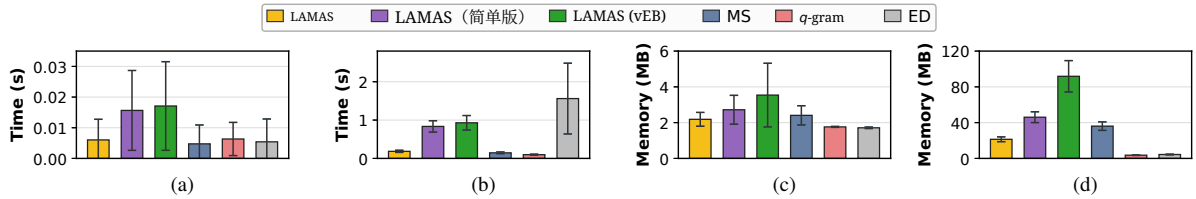


图9：（a）ONT1 和（b）ONT2的运行时间（秒）；（c）ONT1 和（d）ONT2的峰值内存（MB）。

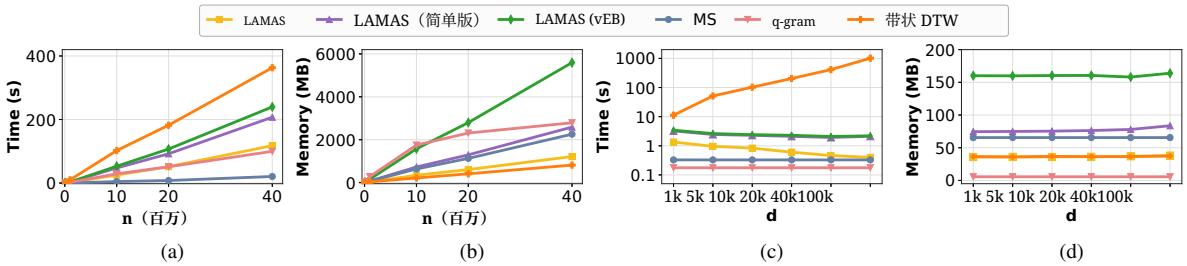


图10：不同 n 下的（a）运行时间（秒）和（b）峰值内存（MB）。不同 d 下的（c）运行时间（秒）和（d）峰值内存（MB）。

在我们的实验中。

图9 显示，在ONT1 和ONT2上，LAMAS 的内存使用量分别比 MS 少 9% 和 41%。此外，在ONT1上，LAMAS 的内存使用量分别比 LAMAS（简单版）和 LAMAS (vEB) 少 20% 和 38%；在ONT2上则少 54% 和 77%。尽管 ED 和 q-gram 距离使用的内存更少，LAMAS 在ONT2上的平均内存使用量仅为 21.4 MB，且速度比 ED 快 8.5 倍。

可扩展性。我们使用人类参考基因组第1和第2号染色体的子串 [Nat13、Nat18]。我们将 LAMAS与其未优化版本、MS以及 q-gram 距离进行比较。由于 ED 的速度慢得多，我们将其省略；在把 DNA字母编码为整数后，我们加入带状动态时间规整作为参考。

n的影响。图10(a)显示，运行时间随 n线性增长。我们固定 $d = 1000$、DTW 带半径 $d = 1000$和 $q = 12$；对于 DNA序列 [Ken02]，通常使用 $[8, 16]$ 中的 q 值。对于 $n \geq 10^5$，LAMAS 始终优于 LAMAS（简单版）和 LAMAS (vEB)。在 $n = 4 \cdot 10^7$, 时，其运行时间比 LAMAS（简单版）低43%，比 LAMAS (vEB)低51%。LAMAS 平均比 q-gram 距离快11.3%，比带状动态时间规整快81.4%。MS 比 LAMAS 快得多，但它不强制执行位移约束，而该约束在上述应用中很有用。图10(b)显示，LAMAS 在缩短运行时间的同时，还减少了内存使用量。在 $n = 4 \cdot 10^7$, 时，其峰值内存使用量分别比 LAMAS（简单版）和 LAMAS (vEB)少53%和78%。它的内存使用量也比 MS 少46%，比 q-gram 距离少56%，但比带状动态时间规整多49%。因此，第4.3节中的改进2不仅提升了时间效率，也改善了空间效率，相较于其他 LAMAS 实现均是如此。

d的影响。图10(c)（$n = 10^6$）显示，将 d 从1000增加到 10^5 可使 LAMAS 的运行时间缩短70%。在 d的六个取值上取平均，LAMAS 分别比 LAMAS（简单版）、LAMAS (vEB) 和带状动态时间规整快 68.6%、70.5%和97.5%。 q-gram 距离最快，其次是 MS；在 $d = 10^5$ 时，MS 仅比 LAMAS 快22%。图10(d)显示，与 LAMAS（简单版）、LAMAS (vEB) 和 MS 相比，LAMAS 分别少使用52%–54%、77%–78%和42%–45%的内存；其内存使用量与带状动态时间规整相当，而 q-gram 距离使用的内存最少。

6 结论

我们介绍了 LAMAS，这是一种具有位置感知能力的知名 MS 原语的推广形式 [CL94], , 并开发了一种计算该原语的时间最优算法，所需的$O(n)$ 时间和空间。我们的实验表明，LAMAS 改进了基于相似度的分类、检索和聚类。我们的 LAMAS 算法实现可在 n 上线性扩展，并且不会因位移容差增大而变慢。

声明

本文的构思、核心思想和研究设计均由作者提出。我们使用 GPT-6 Astra 和 Claude Fable 5.1 改进论文的呈现效果（包括图表的设计）、优化源代码，并协助准备实验。作者对内容、参考文献的正确性和原创性，以及对结果的解读承担全部责任。

参考资料

[ABW15] Amir Abboud、Arturs Backurs 和 Virginia Vassilevska Williams。LCS 及其他序列相似性度量的紧确难度结果。载于 Venkatesan Guruswami（编），IEEE 第56届计算机科学基础年会，FOCS 2015，美国加利福尼亚州伯克利，2015年10月17日至20日，页码59–78。IEEE计算机学会，2015年。doi: 10.1109/FOCS.2015.14.[AHWW16] Amir Abboud、Thomas Dueholm Hansen、Virginia Vassilevska Williams 和 Ryan Williams。用编辑距离及相关问题模拟分支程序：或者说，削去一个多对数因子即可得到下界。载于 Daniel Wichs 和 Yishay Mansour（编），第48届年度 ACM SIGACT 计算理论研讨会论文集，STOC 2016，美国马萨诸塞州剑桥，2016年6月18日至21日，页码375–388。ACM，2016年。doi:10.1145/2897518.2897653.[AP17] Lorraine A. K. Ayad 和 Solon P. Pissis。MARS：利用精炼序列改进多重环状序列比对。《BMC 基因组学》，18:86，2017年。doi: 10.1186/s12864-016-3477-5.[ARK+21] Omar Ahmed、Massimiliano Rossi、Sam Kovaka、Michael C. Schatz、Travis Gagie、Christina Boucher 和 Ben Langmead。用于靶向纳米孔测序的泛基因组匹配统计。《iScience》，24(6):102696，2021年。doi:10.1016/j.isci.2021.102696.[BF00] Michael A. Bender 和 Martin Farach-Colton。重新审视最低公共祖先问题。载于 Gaston H. Gonnet、Daniel Panario 和 Alfredo Viola（编），LATIN 2000：理论信息学，第4届拉丁美洲研讨会，乌拉圭埃斯特角城，2000年4月10日至14日，论文集，《计算机科学讲义》第1776卷，页码88–94。施普林格，2000年。doi: 10.1007/10719839_9.[BF08] Philip Bille 和 Martin Farach-Colton。快速且紧凑的正则表达式匹配。《理论计算机科学》，409(3):486–496，2008年。doi:10.1016/J.TCS.2008.08.042.[BI18] Arturs Backurs 和 Piotr Indyk。除非强指数时间假设为假，否则无法在强次二次时间内计算编辑距离。《SIAM计算杂志》，47(3):1087–1097，2018年。doi:10.1137/15M1053128.[BK15] Karl Bringmann 和 Marvin K¨unnemann。字符串问题与动态时间规整的二次条件下界。载于 Venkatesan Guruswami（编），IEEE 第56届计算机科学基础年会，FOCS 2015，美国加利福尼亚州伯克利，2015年10月17日至20日

2015年10月，页码79–97。IEEE计算机学会，2015年。doi:10.1109/FOCS.2015. 15.[BR11] Ricardo Baeza-Yates 和 Berthier A. Ribeiro-Neto。《现代信息检索：搜索背后的概念与技术》，第二版。培生教育有限公司，英格兰哈洛，2011年。网址：http://www.mir2ed.org/。[CC12] Supaporn Chairungsee 和 Maxime Crochemore。利用最小缺失词构建系统发育。《理论计算机科学》，450:109–116，2012年。doi:10.1016/J.TCS.2012.04.031.[CCF+18] Panagiotis Charalampopoulos、Maxime Crochemore、Gabriele Fici、Robert Mercas 和 Solon P. Pissis。利用缺失词进行无比对序列比较。《信息与计算》，262:57–68，2018年。doi:10.1016/J.IC.2018.06.002.[CFLM25] Davide Cenzato、Giuditta Franco、Zsuzsanna Lipt´ak 和 Alessio Milanese。阈值 q-gram 距离：一种用于基因组序列比较的简单、高效且有效的距离度量。《自然计算》，24(4):973–989，2025年。doi:10.1007/S11047-025-10054-5.[CGK16] Diptarka Chakraborty、Elazar Goldenberg 和 Michal Kouck´y。在低距离范围内嵌入并计算编辑距离的流式算法。载于 Daniel Wichs 和 Yishay Mansour（编），第48届年度 ACM SIGACT 计算理论研讨会论文集，STOC 2016，美国马萨诸塞州剑桥，2016年6月18日至21日，页码712–725。ACM，2016年。doi:10.1145/2897518.2897577.[CGKK18] Moses Charikar、Ofir Geri、Michael P. Kim 和 William Kuszmaul。关于编辑距离估计：比对、降维与嵌入。载于 Ioannis Chatzigiannakis、Christos Kaklamanis、D´aniel Marx 和 Donald Sannella（编），第45届自动机、语言与编程国际研讨会，ICALP 2018，2018年7月9日至13日，捷克布拉格，《LIPIcs》第107卷，页码34:1–34:14。Schloss Dagstuhl - Leibniz-Zentrum f¨ur Informatik，2018年。doi:10.4230/LIPICS.ICALP.2018.34.[CH13] Gideon Creech 和 Jiankun Hu。生成新的入侵检测系统测试数据集：是时候弃用 KDD数据集了。载于2013年 IEEE无线通信与网络会议（WCNC），中国上海，2013年4月7日至10日，页码4487–4492。IEEE，2013年。doi:10.1109/WCNC.2013.6555301.[CKRP26] Panagiotis Charalampopoulos、Tomasz Kociumaka、Jakub Radoszewski 和 Solon P. Pissis。更快的最长公共子串算法。《美国计算机协会算法汇刊》，22(2):19:1–19:47，2026年。doi:10.1145/3774754.[CL90] William I. Chang 和 Eugene L. Lawler。在亚线性期望时间内进行近似字符串匹配。载于第31届计算机科学基础年会，美国密苏里州圣路易斯，1990年10月22日至24日，第一卷，页码116–124。IEEE计算机学会，1990年。doi:10.1109/FSCS.1990.89530.[CL94] William I. Chang 和 Eugene L. Lawler。亚线性近似字符串匹配及其在生物学中的应用。Algorithmica，12(4/5):327–344，1994年。doi:10.1007/BF01185431.[CLZ03] Maxime Crochemore、Gad M. Landau 和 Michal Ziv-Ukelson。适用于无限制评分矩阵的次二次序列比对算法。《SIAM计算杂志》，32(6):1654–1673，2003年。doi:10.1137/S0097539702402007.[CRS20] Peter Christen、Thilina Ranbaduge 和 Rainer Schnell。《敏感数据关联：实用隐私保护信息共享的方法与技术》。施普林格，2020年。doi:10.1007/978-3-030-59706-1。

[Dam64] Fred J. Damerau. 拼写错误的计算机检测与纠正技术。《美国计算机协会通讯》，7(3):171–176，1964年。doi:10.1145/363958.363994.[DBK+19] Hoang Anh Dau、Anthony J. Bagnall、Kaveh Kamgar、Chin-Chia Michael Yeh、Yan Zhu、Shaghayegh Gharghabi、Chotirat Ann Ratanamahatana 和 Eamonn J. Keogh。UCR时间序列档案。《IEEE中国自动化学会自动化学报》，6(6):1293–1305，2019年。doi:10.1109/JAS.2019.1911747.[DYZ+20] Xinyan Dai、Xiao Yan、Kaiwen Zhou、Yuxuan Wang、Han Yang 和 James Cheng。用于编辑距离的卷积嵌入。载于 Jimmy X. Huang、Yi Chang、Xueqi Cheng、Jaap Kamps、Vanessa Murdock、Ji-Rong Wen 和 Yiqun Liu（编），第43届ACM国际信息检索研究与发展会议论文集（SIGIR 2020），线上会议，中国，2020年7月25日至30日，第599–608页。美国计算机协会，2020年。doi:10.1145/3397271.3401045.[FKLR05] Ada Wai-Chee Fu、Eamonn J. Keogh、Leo Yung Hang Lau 和 Chotirat Ann Ratanama-hatana。时间序列查询中的缩放与时间扭曲。载于 Klemens B¨ohm、Christian S. Jensen、Laura M. Haas、Martin L. Kersten、Per-˚Ake Larson 和 Beng Chin Ooi（编），第31届超大型数据库国际会议论文集，特隆赫姆，挪威，2005年8月30日至9月2日，第649–660页。美国计算机协会，2005年。URL: http://www.vldb.org/archives/website/2005/program/paper/thu/p649-fu.pdf.[GBMP14] Simon Gog、Timo Beller、Alistair Moffat 和 Matthias Petri。从理论到实践：紧凑数据结构的即插即用。载于 Joachim Gudmundsson 和 Jyrki Katajainen（编），实验算法——第13届国际研讨会，SEA 2014，哥本哈根，丹麦，2014年6月29日至7月1日。论文集，《计算机科学讲义》第8504卷，第326–337页。施普林格，2014年。doi:10.1007/978-3-319-07959-5_28.[GIM+16] Roberto Grossi、Costas S. Iliopoulos、Robert Mercas、Nadia Pisanti、Solon P. Pissis、Ahmad Retha 和 Fatima Vayani。循环序列比较：算法与应用。《分子生物学算法》，11:12，2016年。doi:10.1186/S13015-016-0076-6.[Gra16] Szymon Grabowski。用于序列相似性问题的新型制表法与稀疏动态规划技术。《离散应用数学》，212:96–103，2016年。doi:10.1016/J.DAM.2015.10.040.[GS18] Omer Gold 和 Micha Sharir。动态时间规整与几何编辑距离：突破二次时间壁垒。《美国计算机协会算法汇刊》，14(4):50:1–50:17，2018年。doi:10.1145/3230734.[GT85] Harold N. Gabow 和 Robert Endre Tarjan。并查集一个特殊情形的线性时间算法。《计算机与系统科学杂志》，30(2):209–221，1985年。doi:10.1016/0022-0000(85)90014-5.[Gus97] Dan Gusfield。《字符串、树与序列算法——计算机科学与计算生物学》。剑桥大学出版社，1997年。doi:10.1017/CBO9780511574931.[HA85] Lawrence Hubert 和 Phipps Arabie。比较划分。《分类学杂志》，2(1):193–218，1985年。doi:10.1007/BF01908075.[Hir75] Daniel S. Hirschberg。计算最长公共子序列的线性空间算法。《美国计算机协会通讯》，18(6):341–343，1975年。doi:10.1145/360825.360861.[HMB24] Christopher Holder、Matthew Middlehurst 和 Anthony J. Bagnall。时间序列聚类弹性距离函数的综述与评估。《知识与信息系统》，66(2):765–809，2024年。doi:10.1007/S10115-023-01952-0.

[JD88] Anil K. Jain 和 Richard C. Dubes。《数据聚类算法》。Prentice Hall，1988年。[JNR05] Anil K. Jain、Karthik Nandakumar 和 Arun Ross。多模态生物识别系统中的分数归一化。模式识别，38(12):2270–2285，2005年。doi:10.1016/j. patcog.2005.01.012.[KCPM01] Eamonn J. Keogh、Kaushik Chakrabarti、Michael J. Pazzani 和 Sharad Mehrotra。大型时间序列数据库中用于快速相似性搜索的降维。《知识与信息系统》，3(3):263–286，2001年。doi:10.1007/PL00011669.[Ken38] Maurice G. Kendall。一种新的秩相关度量。Biometrika，30(1-2):81–93，1938年。doi:10.1093/biomet/30.1-2.81.[Ken02] W. James Kent。BLAT——类似 BLAST 的比对工具。Genome Research，12(4):656–664，四月2002年。doi:10.1101/gr.229202.[Keo02] Eamonn J. Keogh。动态时间规整的精确索引。载于第28届超大型数据库国际会议论文集，VLDB 2002，香港，2002年8月20日至23日，第406–417页。Morgan Kaufmann，2002年。URL: http://www.vldb.org/conf/2002/S12P01.pdf, doi:10.1016/B978-155860869-6/50043-3.[KLA+01] Toru Kasai、Gunho Lee、Hiroki Arimura、Setsuo Arikawa 和 Kunsoo Park。后缀数组中的线性时间最长公共前缀计算及其应用。载于 Amihood Amir 和 Gad M. Landau（编），组合模式匹配，第12届年度研讨会，CPM 2001，以色列耶路撒冷，2001年7月1日至4日，论文集，《计算机科学讲义》第2089卷，第181–192页。施普林格，2001年。doi:10.1007/3-540-48194-X_17.[KSB06] Juha K¨arkk¨ainen、Peter Sanders 和 Stefan Burkhardt。线性工作量的后缀数组构造。《ACM杂志》，53(6):918–936，2006年。doi:10.1145/1217856.1217858.[Lev66] Vladimir I. Levenshtein。可纠正删除、插入和反转的二进制码。Sov. Phys. Dokl.，10(8):707–710，1966年。[LF22] Jindrich Libovick´y 和 Alexander Fraser。神经字符串编辑距离。载于 Andreas Vlachos、Priyanka Agrawal、Andr´e F. T. Martins、Gerasimos Lampouras 和 Chunchuan Lyu（编），第六届自然语言处理结构化预测研讨会论文集，SPNLP@ACL 2022，爱尔兰都柏林，2022年5月27日，第52–66页。计算语言学协会，2022年。doi:10.18653/V1/2022.SPNLP-1.6.[LKLC03] Jessica Lin、Eamonn J. Keogh、Stefano Lonardi 和 Bill Yuan-chi Chiu。时间序列的符号表示及其对流式算法的影响。载于 Mohammed Javeed Zaki 和 Charu C. Aggarwal（编），第8届ACM SIGMOD数据挖掘与知识发现研讨会论文集，DMKD 2003，美国加利福尼亚州圣迭戈，2003年6月13日，第2–11页。美国计算机协会，2003年。doi:10.1145/882082.882086.[LKWL07] Jessica Lin、Eamonn J. Keogh、Li Wei 和 Stefano Lonardi。体验 SAX：一种新颖的时间序列符号表示。《数据挖掘与知识发现》，15(2):107–144，2007年。doi:10.1007/S10618-007-0064-Z.[LMP26] Zsuzsanna Lipt´ak、Francesco Masillo 和 Simon J. Puglisi。匹配统计量综述。《理论计算机科学》，1071:115796，2026年。doi:10.1016/J.TCS.2026.115796.[LPS23] Grigorios Loukides、Solon P. Pissis 和 Michelle Sweering。用于改进文本索引与前K 相似性搜索的双向字符串锚点。《IEEE知识与数据工程汇刊》，35(11):11093–11111，2023年。doi:10.1109/TKDE.2022.3231780.

[LWYH25] Xijuan Liu、Haobo Wei、Peilun Yang 和 Haiyang Hu。CharFormer：一种面向字符的字符串编辑距离注意力网络。《知识系统》，318：113482，2025年。doi：10.1016/J.KNOSYS.2025.113482。[LZL+26] Ling Li、Wiktor Zuba、Grigorios Loukides、Solon P. Pissis 和 Maria Matsangidou。通过索引进行保序模式挖掘与比较。《IEEE知识与数据工程汇刊》，第1–18页，2026年。doi：10.1109/TKDE.2026.3710634。[Mac97] A. Craig MacKinlay。经济学与金融学中的事件研究。《经济文献杂志》，35(1)：13–39，1997年。网址：https://www.jstor.org/stable/2729691。[MEG+23] Santiago Marco-Sola、Jordan M. Eizenga、Andrea Guarracino、Benedict Paten、Erik Garrison 和 Miquel Moret´o。在 O(s) 空间内实现最优间隙仿射比对。《Bioinformatics》，39(2)，2023年。doi：10.1093/BIOINFORMATICS/BTAD074。[MM85] Webb Miller 和 Eugene W. Myers。一个文件比较程序。《软件：实践与经验》，15(11)：1025–1040，1985年。doi：10.1002/SPE.4380151102。[MM93] Udi Manber 和 Eugene W. Myers。后缀数组：一种在线字符串搜索的新方法。《SIAM计算杂志》，22(5)：935–948，1993年。doi：10.1137/0222058。[MMME21] Santiago Marco-Sola、Juan Carlos Moure、Miquel Moret´o 和 Antonio Espinosa。使用波前算法快速进行间隙仿射成对比对。《Bioinformatics》，37(4)：456–463，2021年。doi：10.1093/BIOINFORMATICS/BTAA777。[MN06] Veli M¨akinen 和 Gonzalo Navarro。位置受限子串搜索。载于 Jos´e R. Correa、Alejandro Hevia 和 Marcos A. Kiwi 编，《LATIN 2006：理论信息学，第七届拉丁美洲研讨会，智利瓦尔迪维亚，2006年3月20日至24日，论文集》，《计算机科学讲义》第3887卷，第703–714页。施普林格，2006年。doi：10.1007/11682462_64。[MRS08] Christopher D. Manning、Prabhakar Raghavan 和 Hinrich Sch¨utze。《信息检索导论》。剑桥大学出版社，2008年。网址：https://nlp.stanford. edu/IR-book/pdf/irbookprint.pdf，doi：10.1017/CBO9780511809071。[Nat13] 美国国家生物技术信息中心（NCBI）。智人1号染色体，GRCh38.p12主要组装。RefSeq核苷酸登录号 NC 000001.11，美国国家医学图书馆（美国），马里兰州贝塞斯达，2013年。访问日期：2026年10月。网址：https: // www.ncbi.nlm.nih.gov/nuccore/NC_000001.11。[Nat18] 美国国家生物技术信息中心（NCBI）。智人2号染色体，GRCh38.p12主要组装。RefSeq核苷酸登录号 NC 000002.12，美国国家医学图书馆（美国），马里兰州贝塞斯达，2018年。访问日期：2026年10月。网址：https: //www.ncbi.nlm.nih.gov/nuccore/NC_000002.12。[Nav01] Gonzalo Navarro。近似字符串匹配指南。《美国计算机协会计算调查》，33(1)：31–88，2001年。doi：10.1145/375360.375365。[Pai24] 成对比对基准（pa-bench）。pa-bench数据集，2024年。访问日期：2026年10月。网址：https://github.com/pairwise-alignment/pa-bench/releases/ tag/datasets。[Pea95] Karl Pearson。关于双亲情况下回归与遗传的注记。《伦敦皇家学会会刊》，58(347-352)：240–242，1895年。doi：10.1098/rspl.1895.0041。

[PGR+02] Vasiliki Plerou、Parameswaran Gopikrishnan、Bernd Rosenow、Luıs A. Nunes Amaral、Thomas Guhr 和 H. Eugene Stanley。利用随机矩阵方法研究金融数据中的交叉相关。《物理评论E》，65(6)：066126，2002年。doi：10.1103/PhysRevE.65.066126。[PKS25] Lamprini Pappa、Petros S. Karvelis 和 Chrysostomos D. Stylios。探索丰富多样的基于SAX的方法。《数据挖掘与知识发现》，39(1)：1–78，2025年。doi：10.1007/S10618-024-01075-2。[PT14] Mihai Patrascu 和 Mikkel Thorup。具有最优秩、选择和前驱搜索功能的动态整数集。载于第55届IEEE计算机科学基础年会（FOCS 2014），美国宾夕法尼亚州费城，2014年10月18日至21日，第166–175页。IEEE计算机学会，2014年。doi：10.1109/FOCS.2014.26。[RK04] Chotirat Ann Ratanamahatana 和 Eamonn J. Keogh。利用学习约束提高时间序列分类的准确性。载于 Michael W. Berry、Umeshwar Dayal、Chandrika Kamath 和 David B. Skillicorn 编，《第四届SIAM国际数据挖掘会议论文集》，美国佛罗里达州布埃纳维斯塔湖，2004年4月22日至24日，第11–22页。SIAM，2004年。doi：10.1137/1.9781611972740.2。[ROL+22] Massimiliano Rossi、Marco Oliva、Ben Langmead、Travis Gagie 和 Christina Boucher。MONI：用于查找最大精确匹配的泛基因组索引。《计算生物学杂志》，29(2)：169–187，2022年。doi：10.1089/CMB.2021.0290。[RY98] Eric Sven Ristad 和 Peter N. Yianilos。学习字符串编辑距离。《IEEE模式分析与机器智能汇刊》，20(5)：522–532，1998年。doi：10.1109/34.682181。[SC78] Hiroaki Sakoe 和 Seibi Chiba。口语词识别的动态规划算法优化。《IEEE声学、语音与信号处理汇刊》，26(1)：43–49，1978年。doi：10.1109/TASSP.1978.1163055。[sci] scikit-learn开发者。sklearn.metrics.f1 score。scikit-learn文档。访问日期：2026年10月19日。网址：https://scikit-learn.org/stable/modules/generated/ sklearn.metrics.f1_score.html。[SG02] Alexander Strehl 和 Joydeep Ghosh。聚类集成——一种组合多个划分的知识重用框架。《机器学习研究杂志》，3：583–617，2002年。doi：10.1162/ 153244303321897735。[SKS20] Avi Silberschatz、Henry F. Korth 和 S. Sudarshan。《数据库系统概念》第七版。麦格劳-希尔图书公司，2020年。网址：https://www.db-book.com/。[Spe04] Charles Spearman。论两个事物之间关联性的证明与测量。《美国心理学杂志》，15(1)：72–101，1904年。doi：10.2307/1412159。ˇˇˇˇ[SS17] Martin Soˇsi´c 和 Mile Siki´c。Edlib：一个用于通过编辑距离快速、精确地进行序列比对的C/C++ 库。《Bioinformatics》，33(9)：1394–1395，2017年。doi：10.1093/BIOINFORMATICS/ BTW753。[UBTC06] Igor Ulitsky、David Burstein、Tamir Tuller 和 Benny Chor。用于系统基因组重建的平均公共子串方法。《计算生物学杂志》，13(2)：336–350，2006年。doi：10.1089/CMB.2006.13.336。[UCRa] UCR/UEA时间序列分类存储库。AconityMINIPrinterSmall数据集。访问日期：2026年10月。网址：https://www.timeseriesclassification.com/ aeon-toolkit/AconityMINIPrinterSmall.zip。

[UCRb] UCR/UEA时间序列分类存储库。ACSF1数据集。访问日期：2026年10月。网址：https://www.timeseriesclassification.com/description.php? Dataset=ACSF1.[UCRc] UCR/UEA时间序列分类存储库。ArticularyWordRecognition数据集。访问日期：2026年10月。网址：https://www.timeseriesclassification.com/ description.php?Dataset=ArticularyWordRecognition.[UCRd] UCR/UEA时间序列分类存储库。CharacterTrajectories数据集。访问日期：2026年10月。网址：https://www.timeseriesclassification.com/description. php?Dataset=CharacterTrajectories.[UCRe] UCR/UEA时间序列分类存储库。唐人街数据集。访问日期：2026年10月。网址：https://www.timeseriesclassification.com/description.php?Dataset=Chinatown.[UCRf] UCR/UEA时间序列分类存储库。Crop数据集。访问日期：2026年10月。网址：https://www.timeseriesclassification.com/description.php?Dataset=Crop.[UCRg] UCR/UEA时间序列分类存储库。EOGHorizontalSignal数据集。访问日期：2026年10月。网址：https://www.timeseriesclassification.com/description.php?Dataset=EOGHorizontalSignal.[UCRh] UCR/UEA时间序列分类存储库。FacesUCR数据集。访问日期：2026年10月。网址：https://www.timeseriesclassification.com/description. php?Dataset=FacesUCR.[UCRi] UCR/UEA时间序列分类存储库。Fish数据集。访问日期：2026年10月。网址：https://www.timeseriesclassification.com/description.php? Dataset=Fish.[UCRj] UCR/UEA时间序列分类存储库。GesturePebbleZ1数据集。访问日期：2026年10月。网址：https://www.timeseriesclassification.com/description. php?Dataset=GesturePebbleZ1.[UCRk] UCR/UEA时间序列分类存储库。肉类数据集。访问日期：2026年10月。网址：https://www.timeseriesclassification.com/description.php? Dataset=Meat.[UCRl] UCR/UEA时间序列分类存储库。MixedShapesSmallTrain数据集。访问日期：2026年10月。网址：https://www.timeseriesclassification.com/ description.php?Dataset=MixedShapesSmallTrain.[UCRm] UCR/UEA时间序列分类存储库。PenDigits数据集。访问日期：2026年10月。网址：https://www.timeseriesclassification.com/description.php?Dataset=PenDigits.[UCRn] UCR/UEA时间序列分类存储库。ShakeGestureWiimoteZ数据集。访问日期：2026年10月。网址：https://www.timeseriesclassification.com/description.php?Dataset=ShakeGestureWiimoteZ.[UCRo] UCR/UEA时间序列分类存储库。星光曲线数据集。访问日期：2026年10月。网址：https://www.timeseriesclassification.com/description. php?Dataset=StarLightCurves.

[UCRp] UCR/UEA时间序列分类存储库。UWaveGestureLibrary数据集。访问日期：2026年10月。网址：https://www.timeseriesclassification.com/description. php?Dataset=UWaveGestureLibrary.[UCRq] UCR/UEA时间序列分类存储库。WordSynonyms数据集。访问日期：2026年10月。网址：https://www.timeseriesclassification.com/description.php?Dataset=WordSynonyms.[Ukk85] Esko Ukkonen。近似字符串匹配算法。《信息与控制》，64(1-3):100–118，1985年。doi:10.1016/S0019-9958(85)80046-2。[Ukk92] Esko Ukkonen。利用q-gram和最大匹配进行近似字符串匹配。《理论计算机科学》，92(1):191–211，1992年。doi:10.1016/0304-3975(92)90143-4。[vEB77] Peter van Emde Boas。以低于对数级的时间和线性空间在森林中保持顺序。《信息处理快报》，6(3):80–82，1977年。doi:10.1016/0020-0190(77)90031-X。[VGK02] Michail Vlachos、Dimitrios Gunopulos和George Kollios。发现相似的多维轨迹。载于Rakesh Agrawal和Klaus R. Dittrich（编），《第18届国际数据工程会议论文集》，美国加利福尼亚州圣何塞，2002年2月26日至3月1日，第673–684页。IEEE计算机学会，2002年。doi:10.1109/ICDE.2002.994784。[WF74] Robert A. Wagner和Michael J. Fischer。字符串到字符串校正问题。《《ACM杂志》》，21(1):168–173，1974年。doi:10.1145/321796.321811。[Yah20] 雅虎财经。苹果公司、微软和莫德纳的历史调整后收盘价，2020年。每日观测数据，时间范围为2020年2月3日至6月30日；访问日期：2026年10月22日。网址：https://finance.yahoo.com/。[Yan14] 杨子恒。《分子进化：统计学方法》。牛津大学出版社，2014年。doi:10.1093/acprof:oso/9780199602605.001.0001。[ZHD26] Zhou Zhou、Gang Huang和Laura Dawkins。动态子序列扭曲：一种适用于长时间序列的基于表示的相似性度量。《数据科学与工程》，11(1):260–274，2026年。doi:10.1007/S41019-025-00331-9。[ZZ17] Haoyu Zhang和Qin Zhang。EmbedJoin：基于嵌入的高效编辑相似性连接。载于《第23届ACM SIGKDD国际知识发现与数据挖掘会议论文集》，加拿大新斯科舍省哈利法克斯，2017年8月13日至17日，第585–594页。美国计算机协会，2017年。doi:10.1145/3097983.3098003。[ZZ20a] Haoyu Zhang和Qin Zhang。MinSearch：一种基于编辑距离进行相似性搜索的高效算法。载于Rajesh Gupta、Yan Liu、Jiliang Tang和B. Aditya Prakash（编），《KDD '20：第26届ACM SIGKDD知识发现与数据挖掘会议》，线上会议，美国加利福尼亚州，2020年8月23日至27日，第566–576页。美国计算机协会，2020年。doi:10.1145/3394486.3403099。[ZZ20b] Haoyu Zhang和Qin Zhang。MinSearch：源代码，2020年。访问日期：2026年11月4日。网址：https://github.com/kedayuge/Search。