doi:10.11887/j.cn.202205002 http://journal.nudt.edu.cn

作业名层次化聚类算法预测作业运行时间*

周隆放[1,2],杨文祥[1,3],韩永国[2],张晓蓉[2],喻 杰[1],冯景华[4],张 健[4],
李宇奇[4],鲜 港[1,2],吴亚东[5],王桂娟[2]

(1. 中国空气动力研究与发展中心 计算空气动力研究所,四川 绵阳 621000;
2. 西南科技大学 计算机科学与技术学院,四川 绵阳 621010;3. 国防科技大学 计算机学院,湖南 长沙 410073;
4. 国家超级计算天津中心,天津 300457;5. 四川轻化工大学 计算机科学与工程学院,四川 自贡 643000)

摘 要:预测作业的运行时间有益于提升系统的调度性能,而聚类有助于训练出更好的预测模型。传统的聚类算法很难将相似的作业名聚类,为了将相似的作业更好地聚类,通过分析其组成成分的语义重要性,构建字母-结构-数字的作业名层次化聚类算法。以两台超级计算机的真实数据为例,实验结果发现,应用此算法聚类后的数据训练模型的预测精度相较传统方法有一定的提升,整体预测精度为70%~80%。

关键词:运行时间预测;作业名聚类;机器学习;高性能计算

中图分类号:TN95 **文献标志码**:A **文章编号**:1001-2486(2022)05-013-11

Predicting the job running time with job name hierarchical clustering algorithm

ZHOU Longfang[1,2], *YANG Wenxiang*[1,3], *HAN Yongguo*[2], *ZHANG Xiaorong*[2], *YU Jie*[1], *FENG Jinghua*[4], *ZHANG Jian*[4], *LI Yuqi*[4], *XIAN Gang*[1,2], *WU Yadong*[5], *WANG Guijuan*[2]

(1. Computational Aerodynamics Institute, China Aerodynamics Research and Development Center, Mianyang 621000, China;
2. School of Computer Science and Technology, Southwest University of Science and Technology, Mianyang 621010, China;
3. College of Computer Science and Technology, National University of Defense Technology, Changsha 410073, China;
4. National Supercomputer Center in Tianjin, Tianjin 300457, China;
5. School of Computer Science and Engineering, Sichuan University of Science & Engineering, Zigong 643000, China)

Abstract: Predicting the job running time is beneficial to improve the scheduling performance of the system, and the clustering can help to train better prediction models. Traditional clustering algorithms are difficult to cluster similar job names. In order to better cluster similar jobs, the job name hierarchical clustering algorithm of letter-structure-number was constructed by analyzing the semantic importance of their components. Taking the real data of two supercomputers as an example, the data clustered by this algorithm was used to train the model. The experimental results show that the prediction accuracy of the model is better than that of the traditional method, and the overall prediction accuracy is 70%~80%.

Keywords: runtime prediction; job name clustering; machine learning; high performance computing

在高性能计算领域中,提升系统资源的使用效率长期以来是研究的热点[1-4]。提升硬件资源的使用率和优化作业的吞吐率会在很大程度上减少系统的开销[5-6]。在系统作业调度策略方面,调度策略决定着作业运行的先后次序,从而很大程度地影响着系统作业的吞吐率和资源使用效率[7-8]。传统的作业调度采用先来先服务策略,然而,在先来先服务的策略下,所有作业按照提交时间先后的顺序一次进入等待队列,不同作业请求资源的数目和运行时间都不同,当队首作业不能分配到足够数目的资源时,会等待其他作业运行完成后释放资源。

为了提升系统调度性能,短作业优先策略从等待队列中选运行时间最短的作业运行,这有利于短作业的及时运行,但容易造成长作业处于饥饿状态;时间片轮转策略给每个作业固定的运行时间但是会造成更长的平均等待时间[9];回填调度策略[10-13]在不推迟队首作业正常运行的前提下,选取运行时间最合适的作业抢占空闲资源而运行,该策略能在缩短平均等待时间的同时提升

* **收稿日期**:2021-12-28
基金项目:国家自然科学基金资助项目(61872304,61802320);四川省重点研发资助项目(2022YFG0040)
作者简介:周隆放(1998—),男,四川攀枝花人,硕士研究生,E-mail:longfang_zhou@163.com;
张晓蓉(通信作者),女,讲师,硕士,E-mail:29102239@qq.com

系统的吞吐量,但是作业的预计运行时间难以估计。通常,用户提交作业时会附上该作业的预计运行时间,但是实际上,许多系统的用户并没有被要求去附上作业预估时间甚至用户通常高估作业运行时间。为了回填策略的高效运行,已有大量的算法预估作业运行时间。

近几年机器学习[14-15]的快速发展对于高性能领域的作业时间预测问题有很大的推动。机器学习可以基于大量的历史作业,根据其作业的特征和作业运行时间建立一个作业的预测模型来预测新提交作业的运行时间。传统的预测方法是选取作业的部分特征放入模型中训练[16-17],比如作业的中央处理器(central processing unit, CPU)数量、提交时刻、用户名等。作业名具有作业的语义信息,对描述作业运行特征有重要意义。

聚类是机器学习中提升数据质量的方法之一。作业名由许多数字、字母和特殊符号组成,其结构较为复杂,传统的聚类算法很难将相似的作业名聚类。比如作业名的字母大小写不能区分作业名的相似程度、由相同的字母组成的作业名但字母顺序不同或者字母之间被不同的符号连接、数字仅仅表示值的大小或者时间的先后但无实际的意义。因此仅通过字符串的相似度很难将相似的作业名聚类。用户命名作业的规律:所有的作业名都由字母、特殊字符和数字三个成分组成,用户命名作业往往用字母(英文字母或者拼音或者二者的缩写)表示作业所执行的功能;特殊字符指的是下划线、横线、点和加号等,用户往往用特殊字符连接中间两部分内容,它起到结构化作业名的作用;数字往往被用户用来表示作业的参数或者时间等。根据相似作业的特点,提出字母 - 结构 - 数字(letter-structure-number, LSN)的作业名聚类算法,它总共通过三层聚类作业名:第一层是通过作业名中字母的相似程度聚类;在第一层聚类的基础上,再通过特殊字符的相似程度二次聚类;最后通过数字的相似程度聚类得到最终的作业名聚类结果。

本文采集两个超算中心 2020 年 9 月—2021 年 9 月一年的历史作业数据,抽取数据特征,采用 LSN 作业名聚类算法将相似的作业聚类,采用经典的机器学习模型预测作业的运行时间,包括支持向量回归(support vector regression, SVR)、随机森林(random forest, RF)等。同时采用传统的字符串聚类方法聚类作业,并为之训练模型,通过对比预测精度来评估作业名聚类的效果。

1 相关工作

已经有很多文献提出了作业运行时间预测算法。其中文献[18]在 Mira 和 Intrepid 两台超算上根据用户的提交作业的习惯和特点预测用户新提交作业的运行时间。文献[19]将作业运行时长按照长度区间离散化,然后再综合支持向量回归、贝叶斯岭回归和随机森林回归三种模型,最后通过贝叶斯分类实现运行时间的二次预测;同时该文献还基于径向基的神经网络方法预测作业运行时长。文献[20]通过自定义的特征集合,即模板,预测与之相似的作业的运行时长。文献[21]挖掘半结构的历史作业日志并用隐马尔可夫模型预测作业的剩余运行时间。文献[22]用 K 近邻(K nearest neighbors, KNN)算法预测作业的运行时长并且用留一法和十字交叉验证和统计学的方法为其查找相关系数。文献[23]以用户提交最近的两次作业的平均时长预测作业的运行时间。文献[24]用多项式模型提高作业运行时间预测的精度,它也是通过机器学习的方法,期望通过预测时间从而优化回填调度算法。文献[25]基于 KNN 算法用模板找寻相似的历史作业,并用遗传算法为模型寻找最佳的参数。文献[26]预测特定类型的应用,预测精度有一定的提升。文献[27]提出为避免异常,当预测时间严重偏离应该提醒管理员。文献[28]总结了机器学习的应用,提出了多种表征特征的方法。

作业名由用户命名,用以指征作业的特征,通常表述作业所执行的功能和运行参数等,拥有相同作业名的作业执行的功能类似,所以运行时间可能相近。文献[20]通过作业名将长短作业分离,进一步定义模板训练机器学习模型,但同一作业名下也有可能同时出现长作业和短作业。Last - 2[23]算法简单高效地寻找相同历史作业名的最近两次提交的平均时间作为预测时间,但预测精度到 60% 后存在瓶颈。有大量的文献对数据的新特征进行探索:文献[29]在预测有关原子应用时加入原子的库伦矩阵特征从而提高特定应用的预测效果;文献[30]在输入的特征中加入作业运行的路径从而提高的预测效果。

有大量的研究将作业名按照字符串的距离[25]或者仅保留字母信息的手段将相似的作业名归类,再利用机器学习的手段预测作业的运行时间,但对于命名复杂的作业名,难以通过以上的方法将相似的作业名聚类。本文通过分析用户的

命名习惯,将作业名分为三种组成成分,不同成分的聚类方式不同且成分之间的聚类顺序也不同,最后实验表明所提出的 LSN 聚类方法能有效地将作业名更好地聚类,从而提升了数据质量,进一步地提高了预测的精度。

2　作业时间预测框架

数据来源于两台高性能计算机器:国家超级计算天津中心(天津超算中心)和中国空气动力研究与发展中心(CARDC)。两台机器的数据都是由 Slurm[31] 记录的真实作业数据。数据处理流程如图 1 所示。

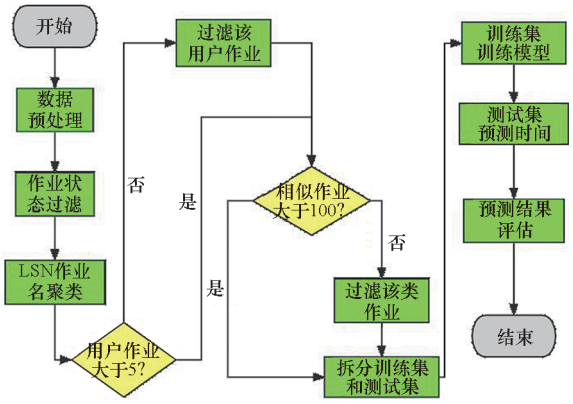


图 1　实验流程概览

Fig. 1　Overview of experiment process

步骤 1:数据预处理阶段是清洗作业日志中的脏数据,比如空值、运行时间为 0 的作业、Linux 脚本文件等少部分对预测时间没有效果的作业,详情见表 1。

表 1　被过滤的作业名称

Tab. 1　Job name being filtered

类别	作业名	说　明
1	sleep、sleep. sh	占节点的作业,无实际意义
2	test. txt、test. exe、test. csh、test. sh	用户用来测试的作业并且没有很强的语义信息
3	run、run. csh、run. sh、run. yh、run. txt、run. exe、run. py	许多用户都会以这样的方式命名程序,该类作业不能指征作业类别
4	sub、sub. sh、main	该类作业不能指征作业类别
5	a、0、1	语义信息太少
6	bash、a. out	该类作业不符合高性能计算作业特征

步骤 2:仅仅保留作业状态为“Completed”的作业。其他状态(“Canceled”“Node _ fail”“Pending”等)的作业无作业完整的生命周期,所以不考虑预测它们的运行时间。

步骤 3:根据提出的 LSN 聚类算法聚类相似的作业名。

步骤 4:过滤小量作业的用户,因为机器学习模型的良好性能得益于大规模的训练数据。小量的作业对提升模型性能没有帮助且不预测小量作业的时间对系统的性能影响较小。

步骤 5:过滤相似作业集合中的小作业集,相似作业集来源于 LSN 作业名聚类的结果。为了取得较为规整的作业数据,保留所有含有 100 条以上作业的集合。

步骤 6:将过滤后的数据用于时间预测,将整体的数据按照 3∶1 的比例分成训练集和测试集,训练集训练时间预测模型,测试集用训练好的模型来预测时间。

3　LSN 聚类算法

3.1　作业名的相似性

传统的时间预测方法是选取能表示作业运行的特征放入机器学习模型中训练,比如作业的 CPU 数量、作业提交的时刻、用户的名称、队列名称等。作业名是作业重要的特征之一,它由用户命名,并且含有作业运行的语义信息。作业名的样例如表 2 所示。

表 2　超算中真实作业名的样例

Tab. 2　Real examples of job names in supercomputer

序号	作业名
1	wmq-v2lack
2	MHorb0S30S_C5SOM. run
3	submit_b4
4	invert_plus2576_tmin0_t
5	sub_DOLPHIN092112. sh

如图 2 所示,集群中的作业名种类繁多,结构丰富。虽然难以归纳作业名的命名规律,但是许多作业名之间彼此相似,相似的作业往往有相似的运行时间。为提高数据的质量,可聚类拥有相似作业名的作业。

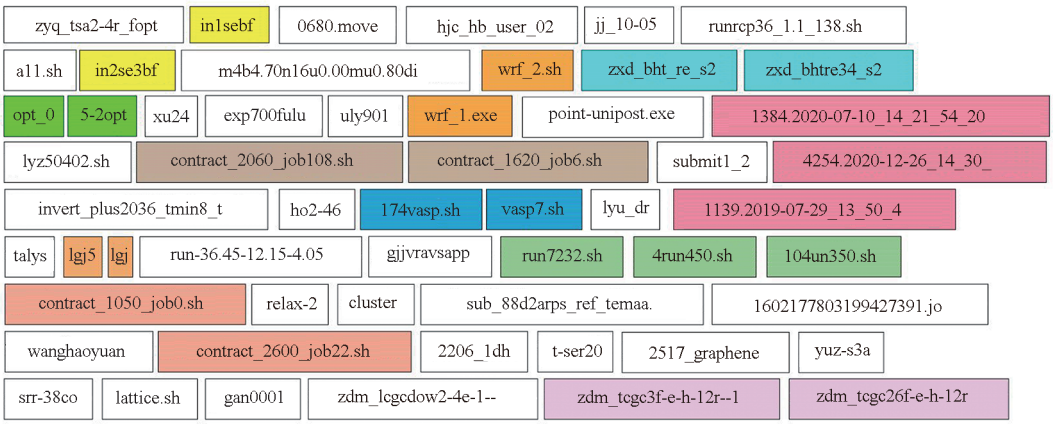


图 2　作业名的样例和彼此相似的作业名

Fig. 2　Sample of job names and job names that are similar to each other

3.2　LSN 聚类详细内容

根据集群中作业名的相似性提出 LSN 聚类算法，目的是将相似的作业名聚类。LSN 聚类的流程如图 3 所示。LSN 算法由字母聚类、结构聚类和数字聚类三个聚类层级组成，如算法 1 所示。作业名集合经过第一层字母聚类后，其中的作业名被聚合成若干个子集 1；每个子集 1 经过第二层结构聚类后被聚合成若干个子集 2；每个子集 2 经过第三层数字聚类后被聚合成若干个子集 3。整个作业名集合经过 LSN 聚类后生成若干个子集 3，聚类的结果输出为所有子集 3 的集合。

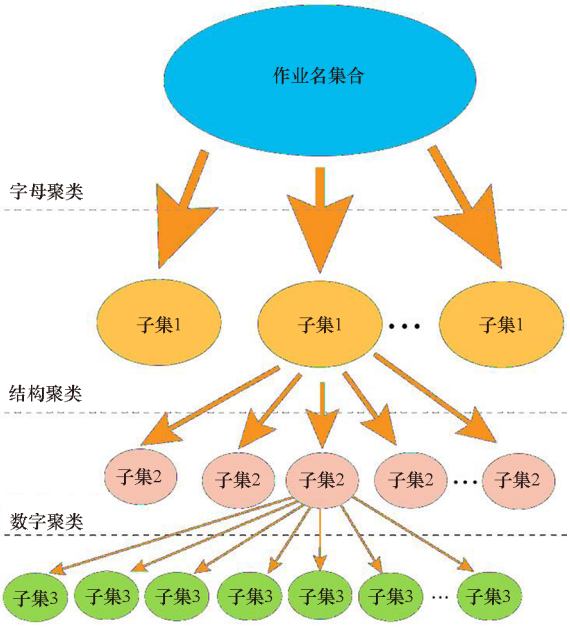


图 3　LSN 聚类算法

Fig. 3　LSN clustering algorithm

3.2.1　作业名的成分特征

作业名实际上是字符的集合，如图 2 所示。作业名的组成成分可被归纳为字母、数字和特殊

算法 1　LSN 作业名聚类算法

Alg. 1　LSN job name clustering

```
已知:作业名集合 Z,结构字符集 S,数字种类集 N
resultSet 设置为空集;
letterSet 设置为空集;
structSet 设置为空集;
numberSet 设置为空集;
while Z is not empty do
  z = Z. pop( ) ;
  zLetterFeature = OnlyContainLetter( z) ;
  ConverUpToLower( zLetterFeature) ;
  if zLetterFeature not in letterSet then
    letterSet. add( zLetterFeature) ;
    letterSet. zLetterFeature. add( z) ;
  else
    letterSet. zLetterFeature. add( z) ;
  end if
end while
foreach l ∈ letterSet do
  foreach s ∈ l. z do
    zStructFeature = FormatByS( s) ;
    if zStructFeature not in structSet then
      structSet. add( zStructFeature) ;
      structSet. zStructFeature. add( z) ;
    else
      structSet. zStructFeature. add( z) ;
    end if
  end foreach
end foreach
numberSet = SelectNoneLetter( structSet) ;
resultSet. add( SelectLetter( structSet) ) ;
foreach s ∈ numberSet do
  foreach n ∈ s. z do
    zNumFeature = FormatByN( n) ;
    if zNumFeature not in resultSet then
      resultSet. add( zNumFeature) ;
      resultSet. zNumFeature. add( z) ;
    else
      resultSet. zNumFeature. add( z) ;
    end if
  end foreach
end foreach
return resultSet
```

字符。作业名的主体为字母和数字,而特殊字符连接字母序列或数字序列,凸显字母或数字序列的层次关系,起着结构化作业名这一重要作用。

作业名的相似性可归纳为字母、结构和数字这三个成分特征的相似性。因为相似作业名的成分特征极为相似,所以 LSN 算法以聚类作业名的成分特征为主要思想,根据各个成分的重要性,构造出字母 - 结构 - 数字的层级聚类算法。

3.2.2　LSN 聚类层级 1:字母聚类

因为作业名中所包含的语义信息很大程度上来源于字母,所以 LSN 层级聚类的第一层为字母聚类。字母聚类的流程如图 4 所示。首先仅保留作业名中的英文字母,紧接着统一大小写,剩余的字母序列为该作业名的特征,最后将含有相同特征的作业名聚合为一类得到若干个图 3 中的子集 1。字母聚类能聚合具有一定相似度的作业名,并且在 CARDC 的数据集中取得了较好的聚类效果。作业名中包含的字母个数越多,字母聚类的效果就越好,但是当作业名中字母个数较少甚至没有字母的时候,子集 1 内容易存在大量互不相似的作业名。如表 3 所示,含有相同的字母在结构上或者参数类别上有较大的差异。

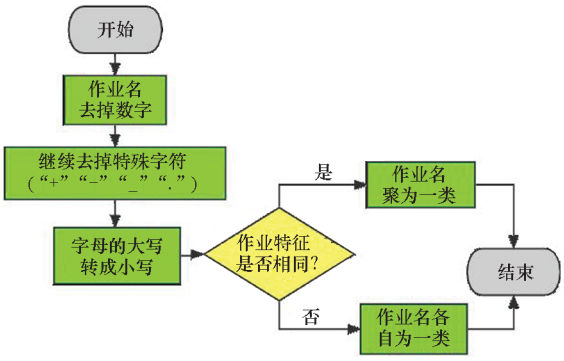


图 4　字母聚类流程

Fig. 4　Letter clustering algorithm flowchart

表 3　作业名的字母聚类算法的结果

Tab. 3　Result of letter clustering algorithm of job names

类别	作业 1	作业 2	作业 3
1	2020 - 11_2_26	0012_23	1.2_0991_8401230
2	Cx_2012	001_cx_992	cX_0.1290_12 - 115
3	Dcb_12	12_D_05c_123_b	012_D_12C_b_00

3.2.3　LSN 聚类层级 2:结构聚类

LSN 聚类的第二层是结构聚类,因为作业名中含有结构化的语义信息。作业名集群中含有的特殊字符如表 4 所示,其中被标红字符常被用户用来连接字母或数字序列,包括“ + ”“ = ”“ - ”“_”和空格(称之为结构字符)。LSN 中由此提出“结构聚类”的概念,因为在拥有相同的字母特征的作业名集合里,作业名之间仍然存在明显的差异,这个差异主要来源于作业名的命名结构,如表 5 所示,这 9 个作业名经过字母聚类后属于 3 个类别,但每一类的内部作业彼此之间结构不同又可以进一步细分成更小子集。结构聚类的方法是将每个字母聚类子集里的作业名以结构字符分节分成若干小节,若小节之间的连接方式相同,则聚为一类。连接方式的相同包括结构字符的数量相同、种类相同、顺序相同。

表 4　作业名的字符组成

Tab. 4　Characteristic composition of job names

类别	名称	内容
1	特殊字符	空格、“ - ”“_”“ + ”“ = ”“ , ”“/”“ * ”“^”“(”“)”“ $ ”“ ! ”“?”“%”“@”
2	字母	a - z、A - Z
3	数字	0 - 9

表 5　有着不同结构的相似作业名

Tab. 5　Similar job names with different structure

类别	作业 1	作业 2	作业 3
1	opt - 4034	201opt_9	52 - 21opt
2	47_ns218_21_5	ns_009	89ns77
3	2co2_560f	cof - 2 - 10	0049_cof

如图 3 所示,每个子集 1 在结构聚类层级上聚类分成若干个子集 2。经过结构聚类后,字母量较少的相似作业名在结构聚类层级能被较好地聚类。但在无字母的作业名集合中,每个子集 2 仍然包含相互之间存在明显差异的作业名。因此 LSN 针对无字母的作业集合,附加额外的聚类层级——数字聚类。

3.2.4　LSN 聚类层级 3:数字聚类

LSN 聚类的第三层级是数字聚类。作业名集合中含有大量的无字母的作业名,该类作业仅仅由数字和特殊字符组成。由于字母不能为该类作业提供丰富的语义信息,LSN 将该类作业名在结构聚类后附加数字聚类。在结构聚类层级上,每个作业名被结构字符分割成若干个子节,LSN 抽

去子节中的非结构特殊字符(“.”除外),所有的子节由数字组成。由于数字的差异无实际意义,因此 LSN 将数字归类。数字可归为 4 个类别——小数、自然数、编号、长序列,如表 6 所示。LSN 判别小数的依据是子节带有“.”;判别编号的依据是子节以 0 开头并且长度大于 2;若非 0 开头但是字符长度大于 3,则属于长序列类别,并且不同的子节长度对应不同的长序列子类别,此时日期就被分为长序列中的四位数的类别,比如 2020;最后 LSN 将不属于上述所有类别的数字类型归为自然数类别。LSN 将每个子节的数字类别都相同的作业名聚类,如图 3 所示,每个子集 2 在数字聚类层级上聚类为若干个子集 3。

若字母仅存在作业名的后缀中,该类作业同样也无语义信息,将后缀名为“.sh”和“.txt”类型的作业名也纳入数字聚类。

表 6 数字类别

Tab. 6 Number category

类别	数值特征	举例
1	小数	7.64、6.905、11.1
2	编号	0010、04300、094
3	自然数	12、2、24
4	长序列	782319238、893243、483294

3.3 聚类结果评估

与传统的聚类算法对比,LSN 算法具有较强的作业名聚类效果。分别以近邻传播(affinity propagation, AP)聚类算法和最大相似长度聚类算法与 LSN 聚类算法进行对比验证。

3.3.1 AP 传播聚类算法

AP 传播算法无须先指定类别的数量,从而达到自动化聚类的效果[32]。它将样本中所有的数据点视为潜在的聚类中心,在表示样本之间相互距离的矩阵中,对角线上的值越大,则对应该位置的样本就越有可能成为聚类的中心。AP 算法迭代所有的点,判断该点和中心点的吸引度、归属度,直到产生所有的高质量的聚类中心停止。采用字符串之间的编辑距离表示两个作业名之间的相似度,距离越大相似度越低。

AP 算法聚类作业名后,每个类别的作业名称都较为相似,但是类别的数量过多且相似的作业名分散在不同的类别里,如表 7 所示。AP 算法的聚类效果较差。

表 7 AP 算法聚类结果

Tab. 7 AP algorithm clustering results

类别	作业 1	作业 2	作业 3
1	Spark – d00 – x05	Spark – d – 10 – x20	Spark – 20 – 0 – x20
2	Spark – d10 – x00	Spark – d – 20 – x10	Spark – d – 10 – x120
3	Spark – d88 – x99	Spark – d88 – x10	Spark – d88 – x98

3.3.2 最大相似长度聚类算法

相似的作业名之间包含相同的子字符串,如表 8 所示。最长公共子序列[33](longest common sequence, LCS)可衡量两个作业名的相似度。

表 8 含有公共子序列的作业名集合

Tab. 8 Collection of job names containing common subsequences

类别	作业 1	作业 2	作业 3
1	xrunPPC_v2_1	xrunPPC_v8_9	xrunPPC_v7_78
2	Cellray_zy_101	Cellray_yx_00	Cellray_yz_91
3	BAI_nlp_v1	BAI_vz_v90	BAI_ZL_turb1

由于作业名的长短差异很大,相似的短作业名的公共子序列也较短。因此用公共子序列的长度与作业名的总长度的比值来判断相似。作业名之间若含有公共子序列并且子序列的长度超过各自作业总长度的 75% 则聚为一类。

LCS 聚类结果与 AP 聚类算法相比有明显的改善,类别的数量有明显的减少,并且每类别的作业名都有公共的子序列。聚类的效果如表 9所示。虽然每个类别的作业名都有公共的子序列,但是每个类别的作业名的相似程度有所降低。

表 9 LCS 聚类结果

Tab. 9 LCS clustering results

类别	作业 1	作业 2	作业 3
1	1XJEW_vX	BJEW_vQ1	RNJEW_v0
2	XZRailBXL0	RailBX1	JARailBX21
3	Star_nlp7	2KKzStar_	Star_new8

3.3.3 LSN 算法聚类结果

LSN 聚类算法由字母聚类、结构聚类、数字聚类三个层级组成。聚类结果如表 10 所示。虽然不同类别的作业名之间存在一定的相似性,但是同一类别下的作业名相似度极高。

表 10　LSN 算法聚类结果

Tab. 10　LSN algorithm clustering results

类别	作业 1	作业 2	作业 3	备注(字母信息)
1	sub1. sh	sub96. sh	sub400. sh	
2	sub_1. sh	sub_4. sh	sub_9. sh	subsh
3	04 – sub. sh	03 – sub. sh	02 – sub. sh	
4	h2o6	4h2o	211h2o1	
5	9 – h2o2 – 3	1 – h2o2 – 2	5 – h2o2 – 2	ho
6	h2o3 – 2	h2o7 – 2	100h2o – 4	
7	23	9	796	
8	0011	0023	0088	无
9	7. 11	5. 233	0. 32	

4　实验结果与分析

4.1　实验流程和评估方法

本实验从 CARDC 的机器得到的历史数据集一共 57 064 条,从天津超算中心得到的数据集一共 67 821 条。只保留完成状态的作业,经过作业状态和作业名的筛选后,CARDC 数据集中共有有效数据 53 332 条以及 2 000 余种作业名,天津超算中心数据集共有有效数据 62 073 条以及 7 285 种作业名。将以上两个有效作业数据集合作为实验的输入。经过 LSN 聚类后,每个作业名都对应一个所属类别的编号。将用户名和作业名类号进行哈希编码。

由于机器学习模型只接受数值型的输入字段,因此在输入的作业字段中,用 UID 代替用户名;相对时间戳代替作业提交时刻,其计算方法如式(1)所示;作业名所属聚类编号代替作业名;用户所申请的 CPU 字段表示 CPU 数量,因为用户申请的 CPU 数目实际上是申请的核数,其数值等于或者小于作业占用 CPU 字段的核数,因此用户申请 CPU 更加能准确表示作业的资源占有数目。

$$t_{\text{relative}\_i} = t_i - \min(t_1, t_2, \cdots, t_n) \tag{1}$$

式中,t_{relative_i}表示相对时间,t_i 表示第 i 个作业的真实时间。

将历史作业数据集按作业提交时间的先后排序,以 3∶1 的比例分成训练集和测试集,训练集中作业的 UID、提交时间、申请 CPU 和作业名类别作为训练特征,作业的实际运行时间作为训练的目标。以预测精度评估预测的准确性,如式(2)所示。Acc 指的是预测精度,t_{predict}是预测的时间值,t_{real}是该作业实际运行时间的真实值,时间单位是 s,精度用百分位表示。

$$Acc = \frac{\min(t_{\text{predict}}, t_{\text{real}})}{\max(t_{\text{predict}}, t_{\text{real}})} \times 100\% \tag{2}$$

4.2　时间预测模型

时间预测的方法采用机器学习的经典预测模型:SVR[34]、决策树[35](decision tree, DT)和 RF[36]。

SVR 是一种监督机器学习方法,它在 n 维输入变量之间构建线性回归函数,计算综合损失函数的值,并且将特征通过核函数映射到高维空间,是非线性回归预测。

DT 是一个树形的模型,它内部的节点在训练的时候根据信息熵分裂,选取基尼系数评判信息增益,是监督机器学习的经典模型之一。

RF 是构造多颗决策树并且最终的输出值由多个决策树共同决定,在很多的预测算法上都体现了其优势性,其也是经典的机器学习方式之一。

4.3　预测结果

基于两台机器 2020 年 9 月—2021 年 9 月一年的真实历史作业数据,采用同样的实验方法,包括数据预处理、作业名聚类、时间预测。以预测精度评估预测的准确性。

4.3.1　作业名有无聚类对比

将未被聚类的作业数据集和经过 LSN 聚类的数据集分别放入模型中训练。在 CARDC 的数据集中一共获得 53 332 条有效数据,其中近 40 000条数据作为训练集,剩下的近 13 000 条作业作为测试集。分别用 SVR、DT 和 RF 的机器学习模型预测作业运行时间,实验结果如图 5 所示。在 CARDC 的数据中,在无作业名聚类的情况下,所有模型的预测精度在 69% ~ 79% 之间;SVR 模型的预测精度较低仅为 69. 2%,DT 和 RF 的预测精度差距较小,都在 78% 左右。在 LSN 作业名聚类的情况下,这三个模型的预测精度均有 1% ~

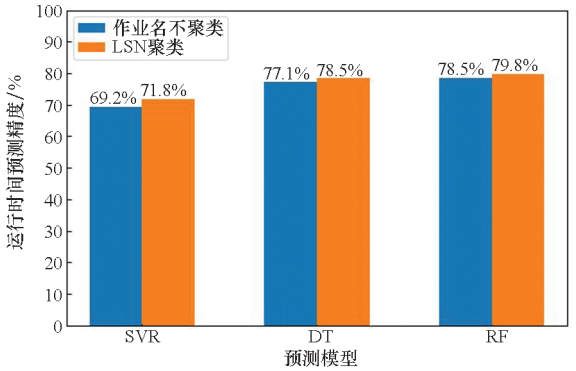


图 5　CARDC 的作业运行时间预测结果

Fig. 5　CARDC's job running time prediction result

3% 的提升,预测精度在 71%~80% 之间。

在天津超算中心的数据集中,一共获得了 62 073条有效数据,同样以 3 ∶ 1 的比例分成近 47 000条作业作为训练集、近 15 000 条作业作为测试集。SVR、DT 和 RF 的模型的预测结果如图 6所示。在无作业名的聚类情况下,三个机器学习模型的预测精度是 62%~74%,天津超算中心的作业预测精度整体较低于 CARDC 的预测精度,主要原因是天津超算中心的作业种类更多、数据量大、用户多,作业的时间较难预测;经过 LSN 聚类后,SVR 预测精度由 62.8% 上升至 74.6%,DT 预测精度由 70.2% 上升至 75.7%,RF 预测精度由 73.5% 上升至 76.8%。相比于 CARDC 的数据,天津超算中心经过作业名 LSN 聚类后时间提升精度更大。天津超算中心的数据中含有7 000 多种作业名而 CARDC 的数据中仅含有 2 000 多种作业名。天津超算中心的作业数据种类更丰富,应用 LSN 算法的聚类效果更明显,时间的预测精度提升更大。

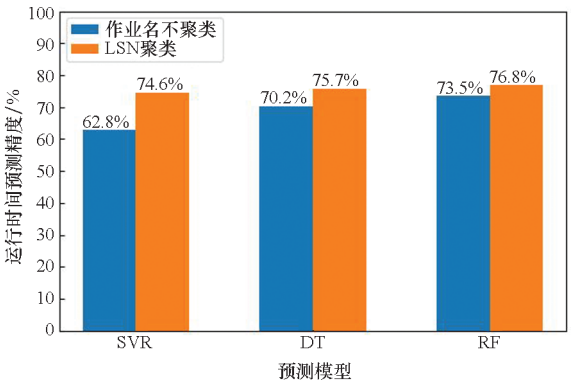


图 6 天津超算中心的作业运行时间预测结果

Fig. 6 Job running time prediction result of Tianjin supercomputer center

在两台机器的数据集中,三个预测时间的模型里,SVR 的预测精度相对低于 DT 和 RF 的预测精度,这主要是因为不同的模型训练的算法不同,SVR 属于非线性回归预测,DT 和 RF 属于分类预测。由于作业的运行时间范围差异较大,从数秒到数十万秒不等,回归算法建立所有数据之间总损失最小的超平面,因此容易在时间范围差异较大的数据集中产生与真实值偏离较大的预测值,从而降低作业时间的预测精度。分类算法的预测值从真实值中产生,因此分类预测更适用于时间范围差异大的数据集合。

4.3.2 不同聚类算法的对比

将 AP 聚类算法、LCS 聚类算法和 LSN 聚类算法得出的三种聚类结果分别放入相同的模型中预测时间。CARDC 的作业的三种聚类算法的预测结果如图 7 所示。在 AP 传播算法的聚类结果中,三个模型的预测精度在 69%~79%;在 LCS 的聚类结果中,三个模型的预测精度在68%~80%;在 LSN 算法的聚类结果中,三个模型的预测精度在 71%~80%。仍然是 DT 和 RF 算法的预测精度整体上较高于 SVR,而且经过 LSN 聚类后的数据集的预测精度在每个模型上都高于经过其他算法聚类后的数据集,主要是由于 LSN 聚类的效果相对于其他聚类算法更好,数据的质量更高。

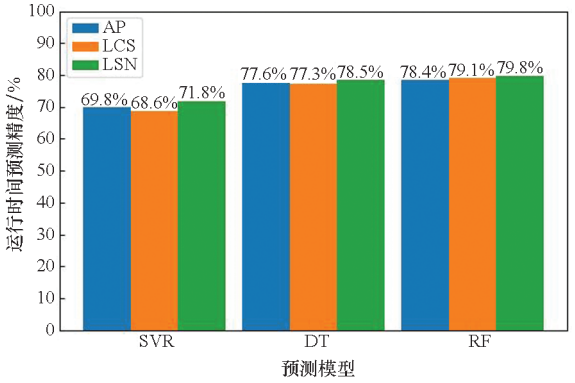


图 7 CARDC 作业不同聚类算法运行时间预测结果

Fig. 7 Runtime prediction results of different clustering algorithms for CARDC jobs

天津超算中心的作业的三种聚类算法的预测结果如图 8 所示。在 AP 传播算法的聚类结果中,三个模型的预测精度在 62%~75% 之间;在 LCS 的聚类结果中,三个模型的预测精度在 70%~74% 之间;在 LSN 算法的聚类结果中,三个模型的预测精度在 74%~77% 之间。不同聚类算法对 SVR 的精度影响较大,对 DT 和 RF 的精度影响较小,但是聚类的质量对时间预测的准确性均有一定的影响。在每个模型中,LSN 聚类算法的数据均比其他两种聚类算法的数据预测精度高,因为 LSN 的作业名聚类效果更好,相应的数

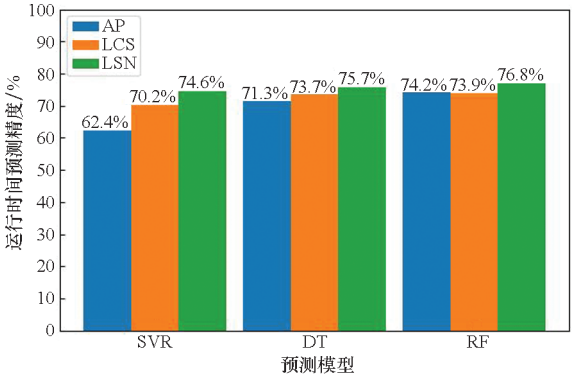


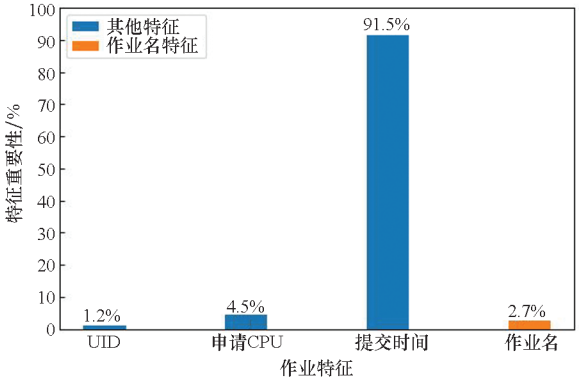
图 8 天津超算中心作业不同聚类算法运行时间预测结果

Fig. 8 Runtime prediction results of different clustering algorithms for Tianjin supercomputer center jobs

据质量更高。于是采取基于 LSN 聚类算法的分类模型预测的结果作为时间预测的最终结果,即 79.8% 的平均精度。

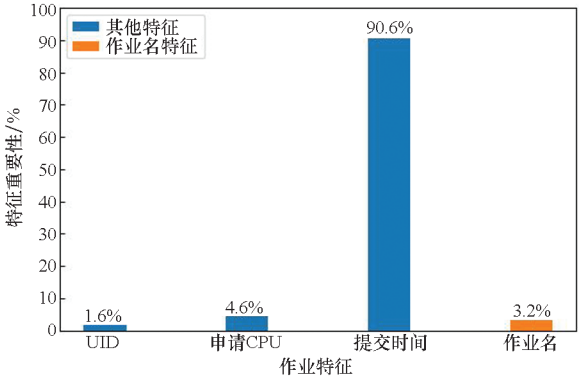
4.3.3　LSN 聚类的特征重要性对比

DT 模型的训练过程是通过基尼系数判断信息增益最大的特征,分裂新的分支而形成的一个树形结构的预测模型。分裂次数多的特征在整个决策树模型的构建过程中起到了相对重要的作用。对比 DT 模型下的各个特征在 LSN 聚类前后的重要性。如图 9(a)所示,在 CARDC 的数据中提交时间特征的重要性达到 91.5%,作业名的重要性仅为 2.7%。如图 9(b)所示,数据经过 LSN 聚类后,所有其他特征的重要性变化均很小但作业名特征的重要性有升高。如图 10(a)所示,在天津超算中心的数据中提交时间特征重要性仍然最高,但相较于 CARDC 的数据其重要性降低,作业名的重要性上升为 12.3%。如图 10(b)所示,经过 LSN 聚类后作业名特征的重要性上升至 15.3%,上升程度明显,因为 LSN 聚类的特征在模型构建过程中的贡献度有所提升。



(a) 作业名聚类前各个特征的重要性

(a) Importance of each feature before job name clustering

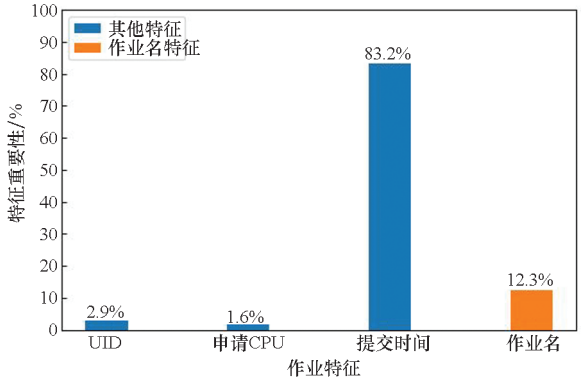


(b) 作业名聚类后各个特征的重要性

(b) Importance of each feature after job name clustering

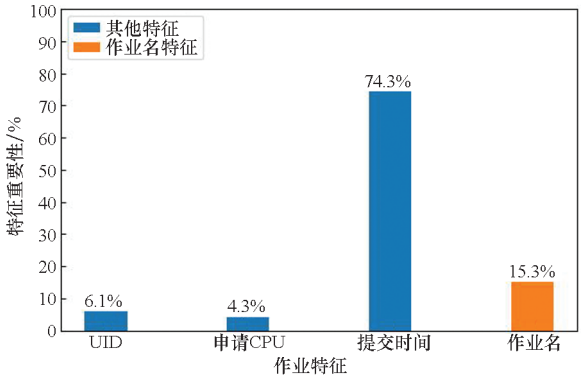
图 9　CARDC 作业的决策树模型的特征重要性

Fig. 9　Feature importance of decision tree model for CARDC job



(a) 作业名聚类前各个特征的重要性

(a) Importance of each feature before job name clustering



(b) 作业名聚类后各个特征的重要性

(b) Importance of each feature after job name clustering

图 10　天津超算中心作业的决策树模型的特征重要性

Fig. 10　Feature importance of decision tree model for jobs of Tianjin supercomputer center

5　结论

本文的数据来自 CARDC 和天津超算中心 2020 年的真实历史数据,通过数据清洗整理出较高质量的作业数据。为提升时间预测效果,提出 LSN 算法聚类作业名。该算法在 CARDC 和天津超算中心的作业名聚类中获得了显著的聚类效果。将预处理后的数据分别放在 SVR、DT 和 RF 的机器学习模型上训练和预测。实验证明经过作业名聚类的数据在这 3 个模型上都取得了时间预测精度的提升,能达到 79.8% 的精度。在未来的工作中将进一步改进作业名聚类算法,深入挖掘作业名包含的语义信息,关联作业名和作业运行时间,提高作业运行时间的预测精度。

参考文献(References)

[1]　KOLLET S J, MAXWELL R M, WOODWARD C S, et al. Proof of concept of regional scale hydrologic simulations at hydrologic resolution utilizing massively parallel computer

resources[J]. Water Resources Research, 2010, 46(4): 312 – 320.

[2] MARO R, BAI Y, BAHAR R I. Dynamically reconfiguring processor resources to reduce power consumption in high-performance processors [C]//Proceedings of Power-Aware Computer Systems, 2000.

[3] HENRIKSSON D, CERVIN A, ÅRZÉN K E. TrueTime: simulation of control loops under shared computer resources[J]. IFAC Proceedings Volumes, 2002, 35(1): 417 – 422.

[4] LEE Y C, ZOMAYA A Y. Energy efficient utilization of resources in cloud computing systems[J]. The Journal of Supercomputing, 2012, 60: 268 – 280.

[5] LIVNY M, BASNEY J, RAMAN R, et al. Mechanisms for high throughput computing [R]. Madison: University of Wisconsin-Madison, 1997.

[6] LAKRA A V, YADAV D K. Multi-objective tasks scheduling algorithm for cloud computing throughput optimization[J]. Procedia Computer Science, 2015, 48: 107 – 113.

[7] KUNTZ P A, CHRISTIE R D, VENKATA S S. Optimal vegetation maintenance scheduling of overhead electric power distribution systems [J]. IEEE Transactions on Power Delivery, 2002, 17(4): 1164 – 1169.

[8] CHO H, RAVINDRAN B, JENSEN E D. An optimal real-time scheduling algorithm for multiprocessors [C]// Proceedings of the 27th IEEE International Real-Time Systems Symposium, 2006.

[9] SIAHAAN A P U. Comparison analysis of CPU scheduling: FCFS, SJF and Round Robin[J]. International Journal of Engineering Development and Research, 2016, 4(3): 124 – 131.

[10] SRINIVASAN S, KETTIMUTHU R, SUBRAMANI V, et al. Characterization of backfilling strategies for parallel job scheduling[C]//Proceedings of International Conference on Parallel Processing Workshop, 2002.

[11] FEITELSON D G, WEIL A M. Utilization and predictability in scheduling the IBM SP2 with backfilling[C]//Proceedings of the First Merged International Parallel Processing Symposium and Symposium on Parallel and Distributed Processing, 1998.

[12] SIVAKUGAN N, RANKINE R M, RANKINE K J, et al. Geotechnical considerations in mine backfilling in Australia[J]. Journal of Cleaner Production, 2006, 14(12/13): 1168 – 1175.

[13] 张吉雄, 缪协兴, 郭广礼. 矸石(固体废物)直接充填采煤技术发展现状[J]. 采矿与安全工程学报, 2009, 26(4): 395 – 401.
ZHANG J X, MIAO X X, GUO G L. Development status of backfilling technology using raw waste in coal mining[J]. Journal of Mining & Safety Engineering, 2009, 26(4): 395 – 401. (in Chinese)

[14] CARLEO G, CIRAC I, CRANMER K, et al. Machine learning and the physical sciences[EB/OL]. (2019 – 12 – 06) [2021 – 11 – 10]. https://arxiv. org/pdf/1903. 10563. pdf.

[15] JORDAN M I, MITCHELL T M. Machine learning: trends, perspectives, and prospects[J]. Science, 2015, 349(6245): 255 – 260.

[16] LI J T, MA X S, SINGH K, et al. Machine learning based online performance prediction for runtime parallelization and task scheduling [C]//Proceedings of IEEE International Symposium on Performance Analysis of Systems and Software, 2009.

[17] GUO J, NOMURA A, BARTON R, et al. Machine learning predictions for underestimation of job runtime on HPC system[C]// Proceedings of Asian Conference on Supercomputing Frontiers, 2018.

[18] FAN Y P, RICH P, ALLCOCK W E, et al. Trade-off between prediction accuracy and underestimation rate in job runtime estimates [C]//Proceedings of IEEE International Conference on Cluster Computing, 2017.

[19] 吴桂宝, 沈瑜, 张文帅, 等. 面向回填优化的作业时长预测[J]. 小型微型计算机系统, 2019, 40(1): 6 – 12.
WU G B, SHEN Y, ZHANG W S, et al. Runtime prediction of jobs for backfilling optimization[J]. Journal of Chinese Computer Systems, 2019, 40(1): 6 – 12. (in Chinese)

[20] NADEEM F, FAHRINGER T. Using templates to predict execution time of scientific workflow applications in the grid[C]//Proceedings of the 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, 2009.

[21] CHEN X, LU C D, PATTABIRAMAN K. Predicting job completion times using system logs in supercomputing clusters[C]// Proceedings of the 43rd Annual IEEE/IFIP Conference on Dependable Systems and Networks Workshop (DSN-W), 2013.

[22] PHINJAROENPHAN P, BEVINAKOPPA S, ZEEPHONGSEKUL P. A method for estimating the execution time of a parallel task on a grid node[C]//Proceedings of A European Grid Conference, 2005.

[23] TSAFRIR D, ETSION Y, FEITELSON D G. Backfilling using system-generated predictions rather than user runtime estimates[J]. IEEE Transactions on Parallel and Distributed Systems, 2007, 18(6): 789 – 803.

[24] GAUSSIER E, GLESSER D, REIS V, et al. Improving backfilling by using machine learning to predict running times[C]// Proceedings of the International Conference for High Performance Computing, 2015.

[25] 许伦凡. 基于历史日志的作业运行时间预测[D]. 绵阳: 中国工程物理研究院, 2019.
XU L F. Job runtime prediction based on historical logs[D]. Mianyang: China Academy of Engineering Physics, 2019. (in Chinese)

[26] 韦建文, 王鎏振, 王一超, 等. 使用机器学习方法预测作业运行时间:以高斯程序为例[C]//CCF 全国高性能计算学术年会, 2021: 519 – 528.
WEI J W, WANG L Z, WANG Y C, et al. Predicting job runtime via machine learning: using Gaussian09 as an example [C]//CCF National Annual Conference on High Performance Computing, 2021: 519 – 528. (in Chinese)

[27] TUNCER O, ATES E, ZHANG Y J, et al. Diagnosing

performance variations in HPC applications using machine learning[C]//Proceedings of International Conference on High Performance Computing, 2017.

[28] SCHMIDT J, MARQUES M R G, BOTTI S, et al. Recent advances and applications of machine learning in solid-state materials science[J]. Npj Computational Materials, 2019, 5: 1 - 36.

[29] BELTRE A, ZAMAN S, CHIU K, et al. Towards run time estimation of the Gaussian chemistry code for SEAGrid science gateway[C]//Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (learning), 2019.

[30] ZHOU L F, ZHANG X R, YANG W X, et al. Prep: predicting job runtime with job running path on supercomputers[C]//Proceedings of the 50th International Conference on Parallel Processing, 2021.

[31] SchedMD. Slurm workload manager version 22.05[CP/OL]. (2011 - 09 - 13)[2021 - 11 - 12]. https://slurm.schedmd.com/documentation.html.

[32] WANG K J, ZHANG J Y, LI D, et al. Adaptive affinity propagation clustering[EB/OL]. (2008 - 05 - 08)[2021 - 11 - 12]. https://arxiv.org/abs/0805.1096.

[33] 郑翠玲. 最长公共子序列算法的分析与实现[J]. 武夷学院学报, 2010, 29(2): 44 - 48.
ZHENG C L. Analysis and implementation on longest common subsequence algorithm [J]. Journal of Wuyi University, 2010, 29(2): 44 - 48. (in Chinese)

[34] BASAK D, PAL S, PATRANBIS C D. Support vector regression [J]. Neural Information Processing Letters & Reviews, 2007.

[35] SAFAVIAN S R, LANDGREBE D. A survey of decision tree classifier methodology[J]. IEEE Transactions on Systems, Man, and Cybernetics, 1991, 21(3): 660 - 674.

[36] BREIMAN L. Random forests[J]. Machine Learning, 2001, 45: 5 - 32.