计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2019,28(10):239−244 [doi: 10.15888/j.cnki.csa.007112]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

系统日志模板提取方法研究①

刘洪歧[1,2], 陈远平[1], 马建化[3]

[1](中国科学院 计算机网络信息中心, 北京 100190)
[2](中国科学院大学, 北京 100190)
[3](福建省龙岩烟草工业有限责任公司, 龙岩 364021)
通讯作者: 刘洪歧, E-mail: liuhongqi@cnic.cn

摘　要: 提取日志模板是处理海量系统日志十分有效的方法. 本文以 Web 系统日志为切入点, 采用基于标签识别树的模板提取方法提取日志模板, 并在其基础上, 研究并完善了其日志预处理和模板表达式生成方法. 针对于系统日志普遍存在的结构复杂问题, 具体采用了基于文本相似度的预处理方法, 实现了日志消息分类; 采用模板最大匹配的方法, 解决了由于日志格式不统一和切词导致的模板匹配度低的问题. 最后, 对本次日志模板提取方法的实验进行了评估, 结果证明该方法的准确率达到 96.4%, 且模板匹配度大幅上升.

关键词: 系统日志; 文本相似度; 日志模板; FP-tree; 标签识别树

引用格式: 刘洪歧,陈远平,马建化.系统日志模板提取方法研究.计算机系统应用,2019,28(10):239–244. http://www.c-s-a.org.cn/1003-3254/7112.html

Research on Extraction Method of System Log Template

LIU Hong-Qi[1,2], CHEN Yuan-Ping[1], MA Jian-Hua[3]

[1](Computer Network Information Center, Chinese Academy of Sciences, Beijing 100190, China)
[2](University of Chinese Academy of Sciences, Beijing 100190, China)
[3](Fujian Longyan Tobacco Industrial Co. Ltd., Longyan 364021, China)

Abstract: Extracting log template is a very effective way to handle massive system logs. In this study, the Web system log is used as the entry point, extracts the log template by using signature tree model. Based on it, we studied and improved the log preprocessing and template expression generation methods. Aiming at the complex structure problem of syslog, the preprocessing method based on text similarity is adopted to realize the classification of log messages. We used the max template matching method to solve the low template matching problem caused by the inconsistent log format and word-cutting. Finally, we evaluate the experiment of this log template extraction method. The results show that the accuracy of the method is 96.4%, and the template matching degree is greatly increased.

Key words: syslog; text similarity; extract template; FP-tree; signature tree

系统日志 (syslog) 是应用系统记录的, 蕴含着大量的信息, 可以记录系统已经发生的行为, 并按照某种规范表达出来. Syslog 可以用于系统异常的诊断、建立系统工作流程等, 然而直接针对海量的非结构化日志进行分析计算是不可行, 应对其做适当的处理, 以加快后续的分析效率. 现有研究工作证明, 提取日志模板是十分有效的日志处理方法, 很多日志分析的相关研究都是依赖于日志模板的. 其相关概念最初是由许巍[1]在其有关于日志分析的文章中提到的, 一条日志模板可以代表一类系统事件, 一个系统产生的海量日志数据都是由少数的日志模板扩展出来的, 日志模板的提取工作是进行其他日志分析工作的基础, 是十分重要且

① 基金项目: 新一代 ARP 试点项目 (XXH13502-01)
Foundation item: New Generation of ARP Pilot Program (XXH13502-01)
收稿时间: 2019-03-22; 修改时间: 2019-04-17; 采用时间: 2019-04-23; csa 在线出版时间: 2019-10-15

有研究意义的. 日志记录通常由常量和变量组成, 也可称为模板词和参数词, 提取日志模板的工作就是提取其中的模板词, 构建模板. 日志模板通常为程序中的 print 定义的语句, 因此基于程序源码解析的日志模板提取方法被广泛的使用, 该方法一般是通过构建程序的抽象语法树 AST, 获取日志输出的相关语句来实现的. 但对于大型系统, 通常包含第三方软件维护的日志, 因此通过源码解析的方式提取日志模板的适应性较差, 且需要具有相关的领域知识[2], 因此本文采用只针对于日志文件的日志模板提取方法. 考虑到已有的实验证明基于标签识别树的方法在准确度和可应用性上都明显优于该领域其余两种常见算法 Statistical Templete Extraction (STE)[3]和 LogSim-ilarity[4], 因此本文选择基于标签识别树的模板提取方法作为具体方法. 该方法应用于实际数据中存在诸多难点: 一是如何对自由形式文本的 syslog[5,6]进行预处理, 使其适合于后续基于标签识别树的模板提取算法; 二是如何在现有模板提取算法的基础上完善模板表达式生成方法, 并提高模板匹配度. 为解决上述问题, 在对 syslog 本身特点和现有领域方法作充分研究后, 本文采用了基于文本相似度的方法对原始日志进行预处理, 并采用模板最大匹配的方法完善了模板表达式生成. 实验证明, 本次采用的日志模板提取方法具有较高的准确度和模板匹配度.

1 系统日志结构分析

系统日志可以记录下系统所产生的所有行为, 并按照某种规范表达出来. 系统日志在安全、信息统计、调试等领域有很重要的作用. 系统日志可以分为操作系统日志、应用系统日志、安全系统日志等等, 由于其服务类型和日志管理方法的不同, 其日志文本形式及其语义语法也存在很大区别. 以 Java 应用系统日志为例, 它有多种可选的日志操作插件, 常见的有 log4j、slf4j、logback 等, 对于大型的 web 应用, 不同功能的日志生成通常是由不同人员管理的, 且通常会含有第三方插件管理的日志模块, 因此日志格式是不统一的. 以使用 log4j 格式的某 Web 应用的系统日志为例, 除开发人员代码中指定的输出内容外, 还有多个可选的打印参数, 如日志优先级、日志输出时间、日志所属时间线程等. 表 1 展示了日志优先级为 INFO 的不同格式的系统日志.

表 1 优先级为 INFO 的系统日志

序号	INFO 类型系统日志
1	[INFO][2018-11-18 00:16:39,256][com.alibaba.druid.pool.DruidDataSource]{dataSource-1} closed
2	2018-11-22 07:53:59,831 INFO [localhost-startStop-1] (DruidDataSource.java:715)-{dataSource-1} inited
3	2018-11-18 01:19:49 [com.alibaba.druid.pool.DruidDataSource]-[INFO] {dataSource-1} closed
4	11-22 07:49:00.013 [startStop-2] INFO o.s.s.c.ThreadTaskExecutor-Shutting down Service 'taskExecutor'
5	22-Nov-2018 07:48:57.840 INFO [main] org.apachel.pause Pausing ProtocolHandler ["ajp-nio-8009"]
6	2018-11-21 08:46:11,898[INFO][c.a.d.p.DruidDataSource]{dataSource-2} closed
7	00:20:04,515 |-INFO in ch.LoggerContext[default] - Could NOT find resource [logback.groovy]
8	2018-11-18 00:16:45,252 [main] 1 INFO - Loading configuration file "./tingyun.properties"

由表 1 可见日志的文本形式是复杂多样的, 要覆盖所有日志格式情况, 提取出用户关心的内容, 需要提前了解日志结构, 本文使用正则表达式的方法将日志内容从每条日志记录中分割出来. 提取表 1 中日志详细内容的对应正则表达式集合如表 2 所示.

表 2 提取日志内容正则表达式

序号	Java 正则表达式
1	\\[INFO\\]\\[.*?\\]\\[.*?\\](.*)
2	\\d+-\\d+-\\d+ \\d+:.*,\\d+ *INFO \\[.*?\\] \\(.*?\\) - (.*)
3	.*\\d+-\\d+-\\d+ \\d+:\\d+:\\d+ \\[.*?\\]-\\[INFO\\] (.*)
4	\\d+-\\d+ \\d+:.*\\.\\d+ \\[.*?\\] INFO +\\w.*? - (.*)
5	\\d+-\\w+-\\d+ \\d+:.*\\.\\d+ INFO \\[.*?\\] \\w.*? (.*)
6	\\d+-\\d+-\\d+ \\d+:.*,\\d+\\[INFO\\]\\[.*?\\](.*)
7	.*INFO in.*?- (.*)
8	\\d+-\\d+-\\d+ \\d+:.*,\\d+ *\\[.*?\\] 1 INFO *- (.*)

日志中的详细信息一般是由模板词和参数词两部分组成的, 模板词内容不随输出时所处的运行状态而改变, 参数词则会根据输出时程序的运行状态进行动态改变, 日志模板可以表征一类系统事件. 虽然日志数据是海量的, 但对应的日志模板通常是很少的, 提取日志模板, 可以从海量的数据中归类出少数的系统运行事件, 能够十分有效的缩减日志分析的维度, 帮助用户更好的理解系统行为.

2 基于 Edit Distance 的日志预处理

基于标签识别树的日志模板提取方法关注的重点是从每一消息类型中提取子类型, 消息类型可以理解

为一组消息特征相近的日志记录, 例如“某服务的状态信息”, “数据的某种操作”等, 虽然这些大的消息类型也可以表征一类日志事件, 但以此来做日志模板的粒度过大, 假设“服务”和“操作”的取值范围很小时, 将消息类型继续细分为“具体服务的状态”和“数据的具体操作”这些子类型, 将可以帮助用户更好的理解系统运行的过程. 日志消息是由模板词和参数词构成的, 考虑到海量日志消息的都是由多个消息类型分别改动少数的参数词扩展而来的, 因此属于同一消息类型的众多日志消息间的文本变动是很小的, 应该具有很高的文本相似度, 而属于不同消息类型的日志记录其之间的模板词和参数词都是大不相同的, 因此彼此之间的文本相似度是很低的. 只有先将海量日志消息进行初步的消息分类, 才能通过基于标签识别树的方法进一步提取日志模板. 本文采用基于 Edit Distance (文本编辑距离) 来衡量日志消息的相似度, 该方法仅在字面层面上进行文本比较[7], 其原理简单且易于实现.

设 S_A、S_B 为两个不同的字符串 A、B, A、B 之间的文本编辑距离即为由 S_A 转换到 S_B 所需要的最少的编辑操作次数, 可选的操作类型有删除、插入、替换 3 种. 假设计算得到的 A、B 的编辑距离为 $Dis(S_A, S_B)$, A 与 B 的字符串长度分别为 $Len(S_A)$、$Len(S_B)$, 则 A、B 之间的文本相似度

$$Sim(S_A,S_B)=1-\frac{Dis(S_A,S_B)}{Max(Len(S_A),Len(S_B))} \quad (1)$$

$Sim(S_A, S_B)$ 的值越大, 则 A, B 间的文本相似度越高.

海量日志中包含的日志模板数是未知的, 要对其进行基于 Edit Distance 的分类, 需要提前确定参照日志. 本文使用日志记录中的第一行日志内容作为基准, 计算其与剩余日志数据的文本相似度, 将结果大于阈值的记录从原数据集中归类并分离出来, 对剩余的日志数据集如此重复迭代计算, 直到数据集为空为止. 这样就能把离线海量的日志数据高效的分类出来. 完整的日志预处理流程如图 1 所示, 假设文本相似度阈值为 τ.

3 基于标签识别树的日志模板提取

基于标签识别树的模板提取方法主要关注从每一消息类型中继续细分出子模板, 它的计算思想来源于关联规则中的 FP-Growth 算法[8], 该算法具有较高的计算效率, 且支持数据的增量更新. 该方法针对上一步日志预处理后得到的日志消息类型簇, 分别提取其子类型模板. 同一消息类型中的日志消息彼此间的文本相似度是很高的, 无法继续通过文本相似度来区分子类型, 但是同一消息类型中属于子模板词的部分应该是高频出现的单词组合, 且可变化范围较小, 参数词则出现频率较低且变化范围广, 这种算法思想正可以映射为 FP-Growth 算法中的频繁项计算, 支持度高的单词组合可以视为子模板词组. 现有的基于标签识别树的模板提取算法主要有模板树的构建和模板树的剪枝两个步骤, 通过这两个步骤可从每一消息类型中分类出具体的子模板词组. 然而要使海量日志数据能够精确匹配日志模板, 需通过提取的模板词组进一步生成模板表达式. 因此本文在原算法的基础上重点针对由于日志格式多样性及文本切词方法等导致的模板匹配度低的问题进行了算法改进, 大幅提升了日志模板匹配度. 下面详细介绍本文采用的日志模板提取方法中的 3 个步骤: 模板树的构建、模板树的剪枝和模板表达式的生成.

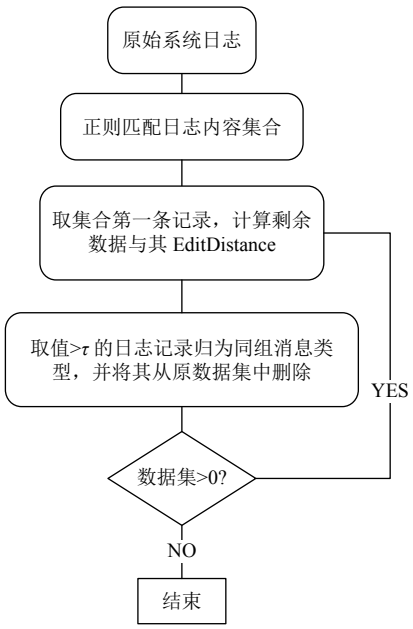

图 1 离线日志预处理流程图

3.1 模板树的构建

模板树的建树过程主要有 3 个步骤:

(1) 切词: 确定特定分隔符, 以此将每条日志记录分割为若干个单词;

(2) 计算词频: 统计在该数据集中每个单词的出现次数;

(3) 按词频建树: 将每条日志中的单词按词频从大到小依次排列, 以“Root”为根节点, 以单词作为其余节

点建 FP-Tree;

在原算法中, 简单将空格作为分隔符, 这种方法对于仅以空格分割的特定格式的日志是有效的, 然而对于多人员开发的日志系统, 其日志格式是复杂的, 不能完全满足这种特定格式, 单以空格作为分隔符, 会导致切词的粒度过大, 计算的准确率下降. 因此本文针对待处理日志本身特点, 依据需要增加分隔符, 改进切词粒度过大的问题. 以表 3 所示的日志记录为例, 使用“ ”、“:”、“;”作为分隔符执行模板树的构建, 构建模板树如图 2 所示.

表 3 模板构建实例

序号	日志内容	按词频重排序
1	流程实例 id: 17122; 工作项 id: 47344; 参与者: 241711-966|241711-6671	流程实例 id, 工作项 id, 参与者, 17122, 47344, 241711-966,|, 241711-6671
2	流程实例 id: 17122; 工作项 id: 47344; 参与者: 241711-6671 | 241711-966	流程实例 id, 工作项 id, 参与者, 17122, 47344, 241711-6671, |, 241711-966
3	流程实例 id: 20630; 工作项 id: 59564; 参与者: 241711-8102	流程实例 id, 工作项 id, 参与者, 241711-8102, 20630, 59564
4	流程实例 id: 17122; 工作项 id: 59561; 参与者: 241711-1021	流程实例 id, 工作项 id, 参与者, 17122, 59561, 241711-1021
5	流程实例 id: 17122; 工作项 id: 59507; 参与者: 241711-8102	流程实例 id, 工作项 id, 参与者, 17122, 241711-8102, 59507
6	流程实例 id: 20631; 工作项 id: 59566; 参与者: 241711-8102	流程实例 id, 工作项 id, 参与者, 241711-8102, 20631, 59566
7	流程实例 id: 15562; 工作项 id: 47060; 参与者: 241711-627	流程实例 id, 工作项 id, 参与者, 241711-627, 15562, 47060
8	流程实例 id: 13303; 工作项 id: 34940; 参与者: 241711-627	流程实例 id, 工作项 id, 参与者, 241711-627, 13303, 34940
9	流程实例 id: 15561; 工作项 id: 42930; 参与者: 241711-627	流程实例 id, 工作项 id, 参与者, 241711-627, 15561, 42930

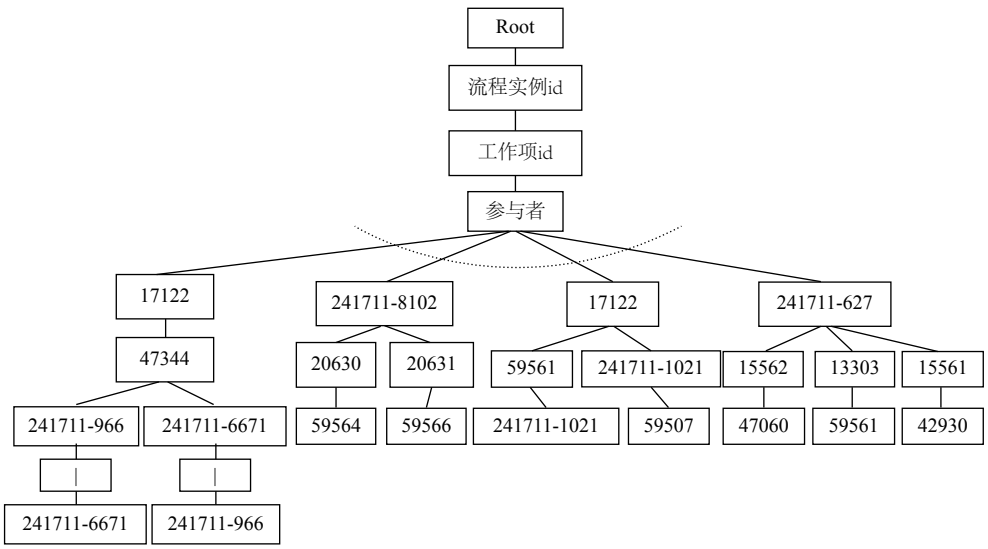

图 2 日志模板树

3.2 模板树的剪枝

对于每组日志类型, 应当只有很少数量的子类型, 而每种类型的参数形式可以是多样的. 对应于日志模板树, 分支较少的节点应为子类型的模板词, 若一个节点的分支数较多, 则该节点的孩子节点很大概率属于参数词, 因此, 该算法中对于构建树中孩子节点数超过设定阈值 k 的节点, 将其所有的子孙节点去除, 将该节点作为模板树的叶子节点, 剪枝后的模板树从根节点到树中叶子节点的每条路径则为子类型模板. 如图 2 中所建的树, 对“参与者”节点的子孙节点进行剪枝, 剩余节点生成的模板组合为“流程实例 id、工作项 id、参与者”, 为子类型模板所包含的关键字.

3.3 模板表达式的生成

初始获得的子类型模板的单词组合是乱序的, 要将原始的日志对应于日志模板, 需要生成模板的正则表达式. 本文采用将获得的单词组合一一对应于所属的原始日志的方式生成日志模板, 注意作为分隔符的字符是不参与对比的, 会在日志模板中以原字符表示,

剩余单词如果属于日志模板则以原字符表示, 否则以任意字符最小匹配的方式——如 java 中以“.*?”表示. 上例中日志索引[1,2,3,4,5,6,7,8,9]都包含模板单词组合“流程实例 id、工作项 id、参与者”, 选取其中一条日志 (如日志索引 1), 按照如上方法, 可生成日志模板表达式: “流程实例 id:.*?;工作项 id:.*?;参与者: .*?.*?.*?”.

可以发现该模板是不合理的, 因为它不能使属于同一模板的所有日志完全与之匹配, 观察模板树可以发现这是由于日志格式不统一和切词导致的. 要获得更通用的模板表达式, 要将分隔符对连续参数组合的影响程度最小, 即参数个数尽可能少, 反映到初始建立的模板树上, 应当选取所属日志中单词个数最少的原始日志作为生成模板表达式的参考文本. 上例中应该选取日志索引 3, 对应生成的模板表达式为: “流程实例 id:.*?; 工作项 id:.*?; 参与者: .*?”, 它可使属于同一模板的日志与之完全匹配.

日志模板提取的完整算法结构如下所示:

算法 1. 日志模板提取算法

输入: 归类的一组日志数据集 $DM=\{M_1, M_2, \cdots, M_n\}$、分隔符集合 *Sep*、剪枝阈值 *k*.
输出: 日志模板表达式.

```
1 Divide each log from DM into a collection of words with separators in Sep
2 Scan the message set DM once.
3 Calculate the support for each word I in DM, let L=map(I, I.support)
4 Create new FP-tree T and the root of T “Root
5 Creat ModelList
6 for each message in DM do
7    Sort its words according to their order in L
8    Let the sorted word list be [p|P],p is the first word in P
9    for each word in P do
10      Call insertNode(p)
11   end for
12 end for
13 for child C in T do
14   if C.child.length > k
15      Delete all the children of C
16      Let the path form C to Root be the list model={Node1, Node2, ⋯, Noden}
17       form the list∈DM that contain C choose Mi that has the Minimum number of words
18      for word in Mi do
19        If word ∉ model
20           Replace the word with.?*
21        end if
22      end for
23      ModelList.add(Mi)
24   end if
25 end for
26 Return ModelList
```

4 实验与评估

本文使用中国科学院某大型分布式信息管理 Web 系统产生的 58 万条 Apach 服务器真实日志记录作为数据集, 使用如上方法进行了日志模板的提取. 实验以 Windows 系统为实验平台, Java 为程序设计语言, MySQL 为数据库进行开发.

4.1 阈值选取

随机选取数据集中的两万条日志记录, 计算其与剩余记录的文本相似度值, 图 3 为其关系折线图. 从图中可以看出, 文本的相似度值从某一点开始骤然下降至非常小的数值, 分析可得该点以后的数据可视为与参考日志属于不同消息类型. 在对大量日志计算转折点后, 发现将文本相似度阈值取为 0.3 较为合适. 对于模板剪枝阈值的选取, *k* 值过大,会提取出多余不必要的日志模板, *k* 值过小则会将本不属于同类的多个子类型归为一类事件. 多次实验证明针对本文的数据集 *k* 取 5 具有较高准确性, 但随这数据集内容的增加, *k* 可能需要依实际情况增大取值.

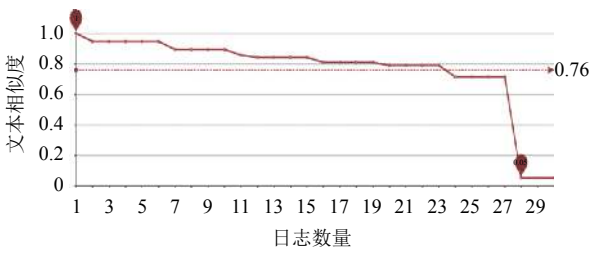

图 3 文本相似度折线图

4.2 实验评估

本次实验从 58 万条 Apache 服务器真实日志数据中提取了 551 条日志模板, 数据源所属 Web 应用的开发人员对于日志消息的分类是十分熟悉的, 具有高可信度, 通过开发人员对日志的手动分类, 获得真实的日志模板. 本文采用 Rand index 方法进行评估, Rand index[9] 方法可用于评估两种数据聚类方法之间的相似性, 将开发人员手动分类的日志模板与实验提取的模板作为 Rand index 算法的输入进行计算, 以此来评估本次实验的准确度. 评估的具体方法为, 从原数据集中预先随

机抽取 100 条数据, 重复在这 100 条数据中随机抽取两条 x 和 y, 定义 4 个评价指标 A、B、C、D 如下:

A: x 和 y 被手动分为一类且实验也被归为一类.

B: x 和 y 被手动分为不同类且实验也被分为不同类.

C: x 和 y 被手动分为不同类但实验被归为一类.

D: x 和 y 被手动分为一类但实验被分为不同类.

使用上述指标, *Rand index* 定义为:

$$Rand\ index = A + B + C + D \tag{2}$$

实验随机抽取了 10 个数据组进行实验, 计算 *Rand index* 值, 实验结果如图 4 所示.

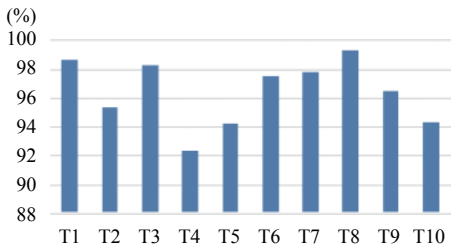

图 4 结果 *Rand index* 计算结果

本实验提取日志模板的准确率平均值达到了 96.4%, 证明基于标签识别树的日志模板提取方法具有较高的可信度和稳定性. 此外, 将实验数据集与提取的模板表达式进行正则匹配, 匹配度为 100%, 证明本文采用的模板表达式的生成方法是合理有效的.

5 结束语

本文实现了基于历史日志文件的日志模板提取方法. 首先根据日志文本特性, 从原始日志中使用基于文本相似度的方法进行迭代计算, 批量完成分类预处理; 然后对分类的日志数据使用基于标签识别树的日志模板提取方法进行模板提取; 最后自主研究并实现了基于上述成果的模板表达式生成工作, 完善日志模板提取算法, 精准的生成模板表达式. 该方法不依赖于源程序相关知识, 具有很好的普适性, 且实验证明该方法同时具有很高的准确性. 本次研究对于基于日志的分析计算具有很重要的意义.

参考文献

1 Xu W. System problem detection by mining console logs [PhD. Thesis]. Berkeley: University of California, 2010.

2 王兆丰, 单甘霖. 一种基于 *k*-均值的 DBSCAN 算法参数动态选择方法. 计算机工程与应用, 2017, 53(3): 80–86. [doi: 10.3778/j.issn.1002-8331.1506-0204]

3 Qiu TQ, Ge ZH, Pei D, *et al*. What happened in my network? Mining network events from router syslogs. Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement 2010. Melbourne, Australia. 2010. 472–484.

4 Kimura T, Ishibashi K, Mori T, *et al*. Spatio-temporal factorization of log data for understanding network events. IEEE INFOCOM 2014-IEEE Conference on Computer Communications. Toronto, ON, Canada. 2014. 610–618.

5 崔元, 张琢. 基于大规模网络日志的模板提取研究. 计算机科学, 2017, 44(S2): 458–462.

6 张曼琪. 基于前缀树的日志模式聚类挖掘算法研究[硕士学位论文]. 上海: 华东理工大学, 2013.

7 陈二静, 姜恩波. 文本相似度计算方法研究综述. 数据分析与知识发现, 2017, 1(6): 1–11. [doi: 10.11925/infotech.2096-3467.2017.06.01]

8 Han JW, Pei J, Yin YW. Mining frequent patterns without candidate generation. ACM Sigmod Record, 2000, 29(2): 1–12. [doi: 10.1145/335191]

9 Rand WM. Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association, 1971, 66(336): 846–850. [doi: 10.1080/01621459.1971.10482356]