**一种基于聚类的系统日志解析算法**

**摘要**

系统日志是软件系统中检查系统状态的重要来源，系统日志中包含的运行时状态报告以及错误信息被广泛地用于系统运维中。随着现阶段软件系统变得日益庞大和复杂，大型软件系统通常会使用日志分析挖掘技术来自动地从系统日志中发掘系统关键信息，日志数据被用于异常检测、根因分析、行为分析等等应用中。日志数据通常是无结构化地文本数据，在适用数据挖掘算法对日志数据进行训练建立模型之前，需要使用日志解析算法进行对原始日志数据进行结构化处理，本文根据日志数据分析挖掘技术的特点和需求，提出一种基于聚类算法的日志解析算法，可以从原始日志数据中提取消息模板和事件序列，同时提出了一种固定深度的树结构模型对消息模板进行存储以实现新日志消息的快速结构化和异常日志消息的检测。通过在特定系统日志数据上的实验证明本文的日志解析算法具有较高的准确性和通用性。

**Abstract**

System log is an important source of checking system status in software system. The runtime status report and error information contained in system log are widely used in system operation and maintenance. With the current software system becoming increasingly large and complex, large-scale software systems usually use log analysis mining technology to automatically mine system key information from the system log. Log data is used in anomaly detection, root cause analysis, behavior analysis and other applications. The log data is usually unstructured text data. Before using data mining algorithm to train and model the log data, it needs to use log parsing algorithm to process the original log data structurally. According to the characteristics and requirements of log data analysis and mining technology, this paper proposes a log parsing algorithm based on clustering algorithm, which can analyze the original log data from the number of original logs At the same time, a fixed depth tree structure model is proposed to store the message template to realize the fast structure of new log messages and the detection of abnormal log messages. Through the experiments on the specific system log data, it is proved that the log parsing algorithm in this paper has high accuracy and generality.

**1 引言**

系统日志是在系统实时运行时记录硬软件问题的文本信息，系统日志可以被用来监视系统运行时发生的事件信息，比方说事件的时间戳、一个请求的IP地址或者一个异常任务的状态。系统日志蕴含的丰富信息可以帮助系统开发人员和维护人员更好地理解系统行为并在生产过程中检测和定位系统异常。除此以外，对系统日志的分析和挖掘技术也被应用于金融和政府机构中，比方说，交易系统日志被用来进行异常交易的检测，电力系统日志被用来进行安全隐患的查找。

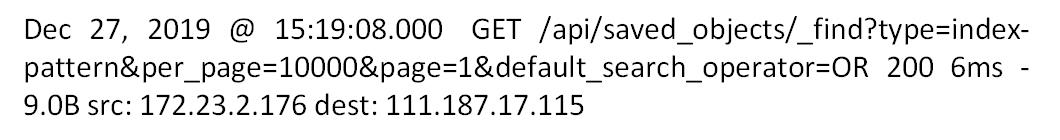
随着现阶段软件系统规模和复杂度的日益增加，系统日志规模也变得更加庞大，大型软件系统会按照每小时10gb的速度产生日志。传统的人工方式对这些日志信息进行分析是非常不切实际的，因此，全自动的基于日志数据的分析挖掘技术是迫切需要的。比较典型的日志挖掘技术包括异常检测、行为分析、程序验证、根因分析等等。这里需要引用参考文献。另外，由于现阶段算法和算力的发展，越来越多的开发人员选择使用机器学习和深度学习技术进行日志挖掘。然而，由于开发人员在记录系统日志为了简单和方便会使用简单文本，因此系统日志通常是无结构的文本数据，在使用数据挖掘算法对系统日志建立模型训练之前，需要使用日志解析算法将原始的日志数据结构化。

图1 一条典型的日志消息

一条典型的日志消息，如图1所示，记录了一件特定的系统事件，包括多个部分：时间戳（记录了事件发生的具体时间），源IP地址和目的IP地址，以及原始的消息内容。日志解析技术一般是针对日志事件中的消息部分，即图1中的“GET /api/saved\_objects/… 200 6ms - 9.0B src: 172.23.2.176 dest: 111.187.17.115”。消息部分通常可以分为常量部分和变量部分，常量部分是固定不变的文本内容，通常定义了一个消息模板的事件类型，变量部分显示了系统的运行时信息，比方说状态的值和参数（IP地址、持续时间、文件路径等等），变量部分在不同的日志中会有不同的值。日志解析算法的目标时自动化地从日志消息部分中提取常量部分和变量部分，并将每一条日志转化为一个特定的事件，比方说图1所示例子中的常量部分为“GET \* \* - 9.0B src: \* dest: \*”,变量部分我们用通配符替代，常量部分就代表了一个特定的事件。在本文中我们使用消息模板/事件类型来指消息部分中的常量部分。

日志解析技术是日志挖掘的关键一环。传统的日志解析算法包括正则表达式匹配、基于源代码的方式，或者基于规则式的方式。现代软件系统的迭代和发展速度过于迅速，一般系统中定义日志消息的源代码很难获得，并且日志的规则需要对系统非常熟悉的专业人士来制订，当系统发生变化时，规则也需要同时发生改变，这是比较费时费力的，因此传统的日志解析算法并不能适用于现在的软件日志系统。

一般来说，日志中消息部分的常量部分是由系统中定义日志输出的代码决定的，通常同一条消息模板产生的日志消息都非常相似。聚类算法的思想即是通过比较实例之间的距离，并将相似的实例聚为一类，本文利用这种思想，提出了一种基于聚类的日志解析算法，并使用一种固定深度的树结构模型来存储消息模板，以实现对新的日志消息快速解析和异常日志的检测。

本文的组织如下：第二部分本文将会给出相关学术工作以及符号定义，第三部分介绍一种基于聚类的日志提取算法，第四部分给出本文实验在特定系统日志数据上的实验结果，第五部分为总结与展望。

**2** **问题描述与相关工作**

2.1 问题描述

系统日志一般是独立的文本数据，文本内容记录了系统中发生的事件，包括消息部分和一些事件属性，事件属性包括时间戳、消息级别等等。消息部分时由源代码定义的，比方说下面的一行代码：

可以产生下面的日志消息：

这些消息可以形成一个消息模板/事件类型，将其中的变量部分用通配符表示，则该条消息模板可以表示为：

大部分情况下，一条消息模板对应一条打印语句，日志解析就是从大量的原始日志数据中提取消息模板，这样一条日志就对应一个事件类型，按照时间顺序排列即得到事件序列。

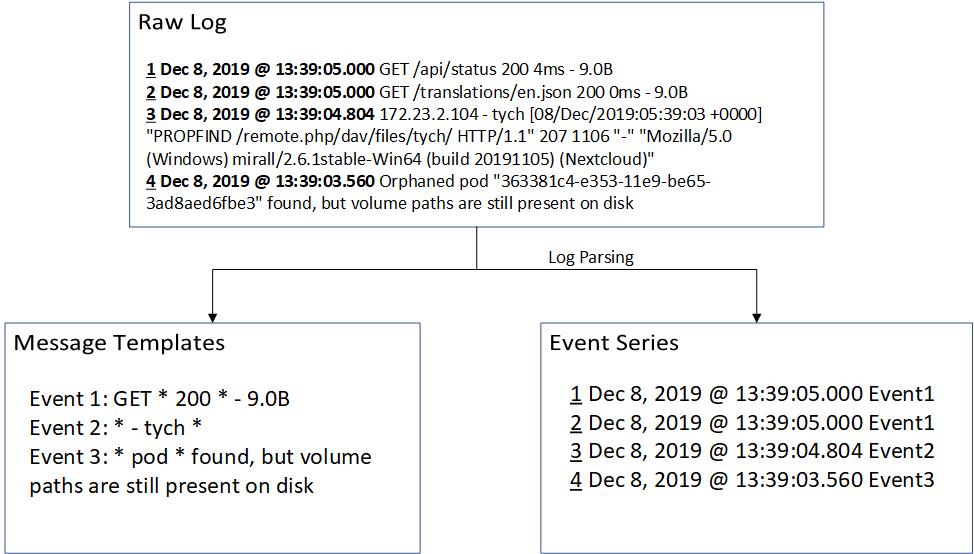
图2展示了一个日志解析工具的主要功能，其中原始日志数据来源与在线实训平台功夫编程kfcoding，由图2可知，原始的系统日志时非结构的，每一条日志都包含时间戳、类别、消息内容，真实情形下，系统在一天中会产生成千上万条类似这样的日志信息，日志解析将会读取原始日志消息并生成消息模板和事件序列。

图2

2.2 相关工作

2.3 符号定义

我们在这个小节给出基于聚类的日志解析算法的相关符号定义。

定义1 事件日志 定义事件日志为记录系统内发生的事件或对应用程序跟踪记录的文本。

定义2 事件 事件为事件日志中一行独立的文本，详细地说明了系统或应用程序发生的一次事件。

定义3 消息 消息是指事件中除去时间戳、类型、标签等信息，只记录具体事件信息的文本序列。

定义4 标记 消息部分中被分隔符划分的独立的单词为标记。一般情况下，分隔符为空格。

定义5 消息长度 消息部分中标记的个数。

定义6 消息模板/事件类型

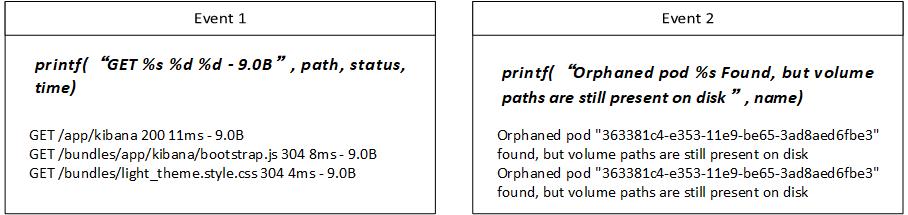
基于聚类的日志解析算法

3.1 算法概述

通过对大量的日志数据进行分析发现，由相同的消息模板产生的日志通常都比较相似，这与它们都有相同的常量标记有关，聚类算法的主要思想是通过比较实例之间的相似性，将相似的实例聚为一类，这与日志解析的目的类似，日志解析是将相似的日志消息（由同样的消息模板产生的日志消息）聚为一类并从中提取公共部分作为消息模板。另外由于聚类算法会涉及到大量的相似度计算，因此我们在聚类算法的前一步会首先对日志数据进行处理以减少时间复杂度。

根据上文的分析，本文基于聚类算法的日志解析算法通过四个步骤实现日志解析的功能，预处理会统计每个消息的消息长度，并使用简单的正则表达式替换消息中明显的标记变量，将相同长度的消息分为一组，在同一组中使用聚类算法将同组中相似的消息聚为一类，由于经过聚类后，相同类别的消息也许还是由不同的消息模板产生的，在聚类之后会在同类中进行分组，使得每一个组都对应一条独立的打印语句，在每一组中生成消息模板/事件类型。

3.2 事件分组和标记变量替换

消息长度定义为消息中独立的标记的数量，我们认为，由相同的消息模板产生的日志消息具有相同的消息长度，如图中所示，事件1和事件2消息长度是不同的，将相同的消息长度分为一组的另一个主要目的是，聚类算法会进行大量的距离计算，预处理可以减少聚类的时间复杂度，并且有利于提升整体算法的运行效率。

综上所述，这一步的主要做法是统计日志数据集中每一条消息的消息长度，使用正则表达式将消息中明显的标记变量进行替换，将相同长度的消息分为一组，形成MessageMaps，具体的算法以伪代码的形式如表1.

|  |
| --- |
|  |
|  |

3.2 聚类

我们在上一步得到的每一个MessageMap中使用聚类算法将相似的消息聚为一类，聚类中比较重要的是距离的度量，我们在这里使用加权的字符串编辑距离，具体定义如下：

其中

本文对于两个不同的消息，会首先计算它们的加权编辑距离，如果小于阈值，则它们之间具有联系，我们将所有两两之间具有联系的消息分为一个组MessageGroup。

本文在这里给出一种阈值的确定方法，通过计算每对消息之间的加权编辑距离，我们可以得到训练集中所有对的加权编辑距离的集合，对它们进行k-means聚类，选定k为2，选定其中类中心值比较小的类中最大加权距离作为阈值。

表 给出了聚类的完整算法MessageCluster，

3.3 类中分组

经过预处理和聚类两个步骤，在一个MessageCluster中的消息也可能出自不同的消息模板，因此我们在这一步进行对MessageCluster的分组。

这一步的主要思想为，对于一个MessageCluster，N为其中消息的消息长度，设其中消息的公共标记有M个，将每条消息划分为M+1个位置，在每一个位置上，我们统计共有多少种不同的标记序列，用V表示，一般来说，V的值越大说明该位置越不可能式标记常量，通过引入阈值，当V超过阈值，我们认为这个位置为标记变量，并且不再进行划分操作，当V不超过阈值时，则该位置为标记常量，我们将这个MessageCluster划分为V+1组。

3.4 消息模板生成

我们认为一个MessageGroup为一个消息模板生成的消息，将其中消息的公共部分作为标记常量，非公共部分为标记变量，将标记变量用通配符表示可得到消息模板描述。

实验结果与分析

本文的日志数据来源于在线实训平台功夫编程kfcoding,功夫编程是采用微服务架构思想构建的，旨在满足三种不同的业务场景下不同用户的需求，第一中业务场景为个人学习场景，在该场景下，个人用户可以通过该平台学习和发现最新的前沿IT技术，如云计算、大数据、人工智能、机器学习、容器技术等等。用户不仅可以看到教程书籍，还能够在平台上进行实践。与此同时，用户还可以自己创建/编辑/删除书籍，即具有书籍管理功能。

第二种业务场景为学校教学场景。在该场景下，学校教师可以通过该平台开设课程，上传课件等资源，并布置作业。相应的，选修该课程的学生可以在平台上进行课程学习，边看课件边动手操作，并可以通过该平台提交作业。

第三种业务场景为企业实训活动场景。在该场景下，实训活动举办方（企业/组织）可以定制自己的交互式产品文档，采用平台提供的定制化的统一配置环境，举办实训活动或团队培训。而用户通过参与活动即可以在真实的使用环境中快速学习或体验相关产品。

功夫编程的整个系统架构如图所示：

功夫编程会以大概每小时4000条日志的速率产生日志事件，我们搜集了其中一个月的日志数据进行日志解析算法，这一个月的日志数据作为日志训练集，经过步骤一到步骤四，无结构的原始日志数据会解析出消息模板和事件序列，我们采用固定深度的消息模板树来存储消息模板，以实现对新的消息快速解析和异常日志的检测。

我们拿出典型的日志作为展示算法具体的运行效果，如图

