一种复合型日志的模板提取方法

作者：吴其 马严 黄小红 丛群
北京邮电大学--网络技术研究院

目录

- 研究背景
- 研究内容
- 实验过程

一、研究背景

研究背景与现状
解决方案

研究背景与现状

背景

- 现有的算法无法处理日志中包含Json或Key-Value等类型数据的日志
- 如：
  - client response: { "view": "policy_view_23" }
- 会被处理为：
  - client response: { "view": "policy_view_*" }
- 而实际它应被处理为：
  - client response: { "view": "*" }

现有的算法

- Drain算法
- SHISO算法
- Spell算法
- ……

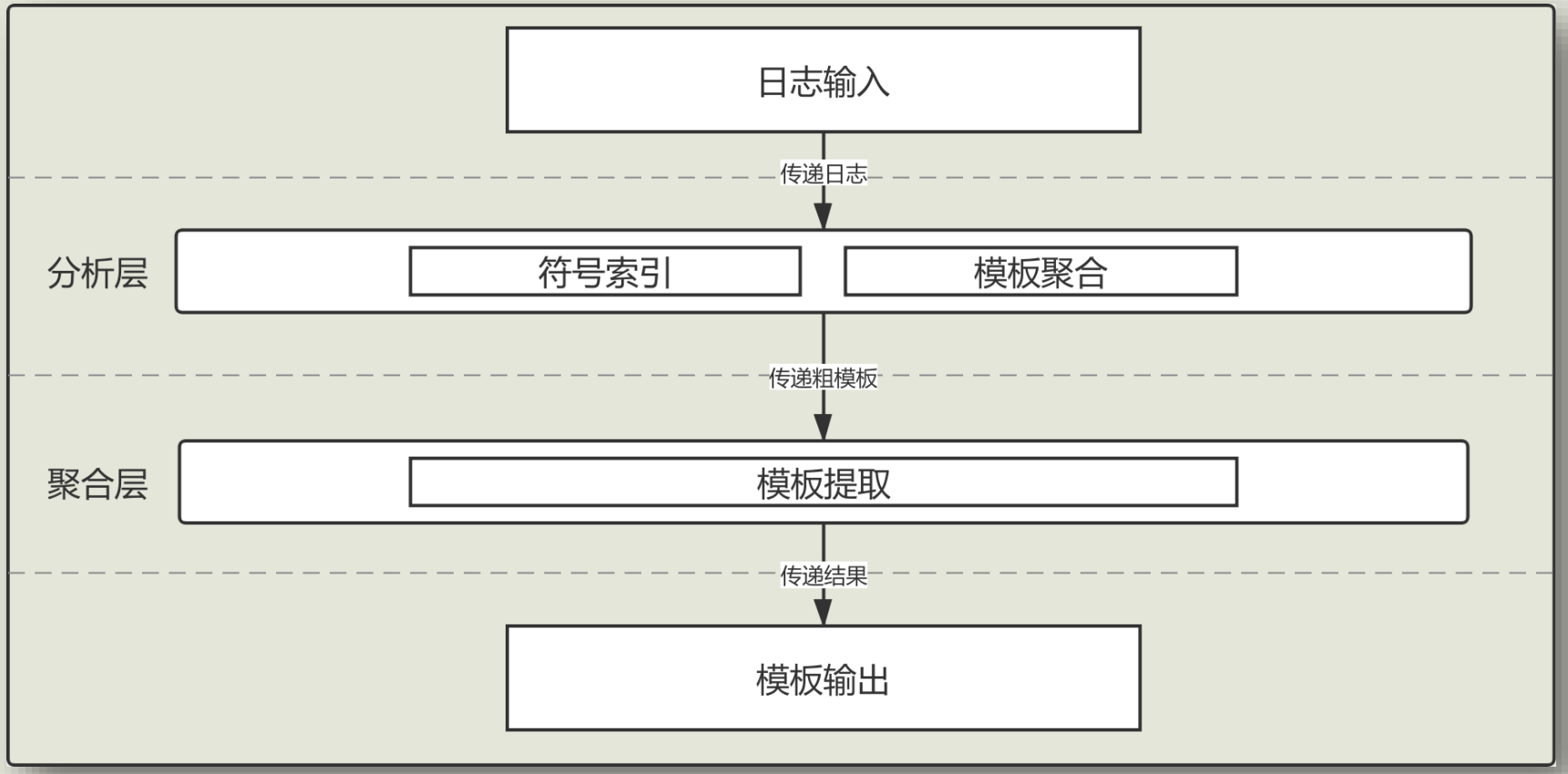
解决方案

- 建立符号索引，先合并结构相同的日志
- 改进相似度计算的算法，保证算法对微小差异的敏感性
- 设计BMerge算法实现对日志模板的无损合并

二、研究内容

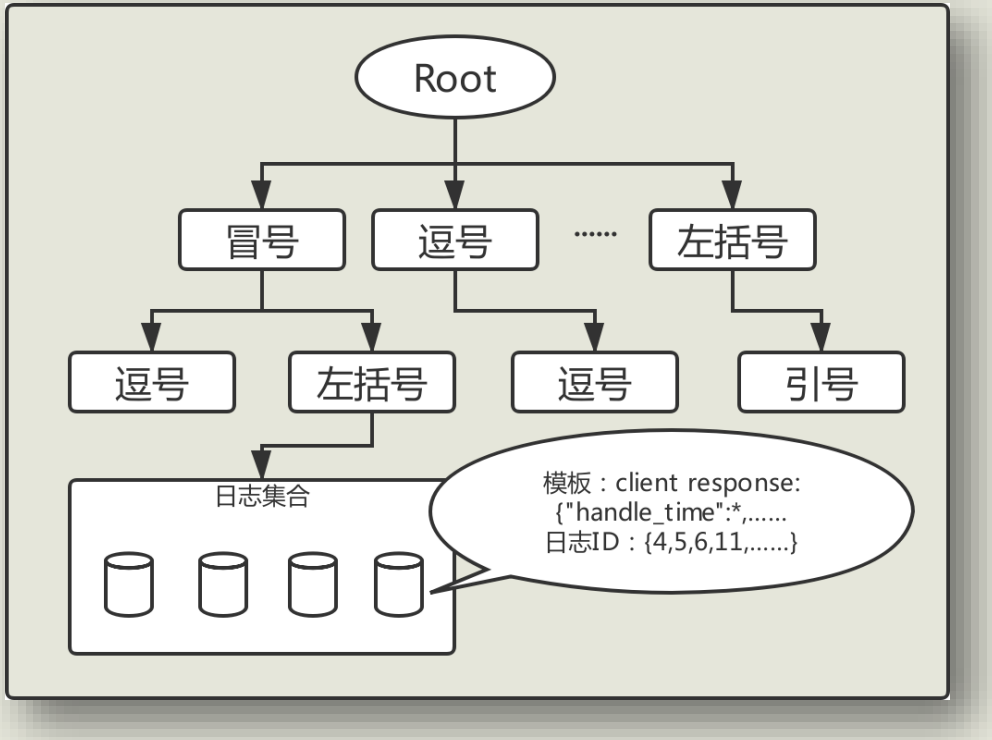
算法流程
算法实现

算法流程



算法实现（一）

- 符号索引
  - 以符号对日志进行集群划分
  - 具有相同符号结构的日志将被划分到同一个集群
  - 如：
    - client response: { "handling_time" : 255 }
    - client response: "handling_time" : 225
  - 两条会被分入到不同的集群



算法实现（二）

- 模板聚合
  - 使用Drain算法对每个集群进行处理，得到初始的模板

```
EventId,EventTemplate,Occurrences
f764dfec,view policy_view_<*>: query: <*> IN <*> <*> (<*>),72
876c6c40,"{ ""handling_time"": <*>, ""client"": ""<*>"", ""client_port"": <*>, ""qname"": <*> ""view"": ""policy_view_<*>"", ""type"": ""A"", ""result"": ""OK"", ""ret"": [ <*> <*> <*> <*> <*> ] }",2
0319b129,"{ ""handling_time"": <*>, ""client"": ""<*>"", ""client_port"": <*>, ""qname"": <*> ""view"": ""policy_view_<*>"", ""type"": <*> ""result"": ""NXDOMAIN"" }",30
43619e64,"{ ""handling_time"": <*>, ""client"": ""<*>"", ""client_port"": <*>, ""qname"": <*> ""view"": ""policy_view_<*>"", ""type"": <*> ""result"": ""OK"", ""ret"": [ <*> <*> <*> <*> <*> <*> <*> <*> <*> <*> <*> <*> ] }",4
d58ca104,"{ ""handling_time"": <*>, ""client"": ""<*>"", ""client_port"": <*>, ""qname"": <*> ""view"": ""policy_view_<*>"", ""type"": <*> ""result"": ""OK"", ""ret"": [ <*> <*> <*> <*> <*> <*> <*> <*> ] }",3
f07a9f02,"{ ""handling_time"": <*>, ""client"": ""<*>"", ""client_port"": <*>, ""qname"": ""."", ""view"": ""policy_view_<*>"", ""type"": ""NS"", ""result"": ""OK"", ""ret"": [ <*> <*> <*> <*> <*> <*> <*> <*> <*> <*> <*> <*> <*> <*> ] }",10
a8487019,"{ ""handling_time"": <*>, ""client"": ""<*>"", ""client_port"": <*>, ""qname"": ""<*>-courier.sandbox.push.apple.com"", ""view"": ""policy_view_<*>"", ""type"": ""A"", ""result"": ""OK"", ""ret"": [ ""<*>-courier.sandbox.push.apple.com. <*>\tIN CNAME <*>.courier-sandbox-push-apple.com.akadns.net."", ""<*>.courier-sandbox-push-apple.com.akadns.net. <*> IN CNAME us-sandbox-courier-<*>.push-apple.com.akadns.net."", ""us-sandbox-courier-<*>.push-apple.com.akadns.net.\t300 IN A <*>"", ""us-sandbox-courier-<*>.push-apple.com.akadns.net.\t300 IN A <*>"", ""us-sandbox-courier-<*>.push-apple.com.akadns.net.\t300 IN A <*>"", ""us-sandbox-courier-<*>.push-apple.com.akadns.net.\t300 IN A <*>"", ""us-sandbox-courier-<*>.push-apple.com.akadns.net.\t300 IN A <*>"", ""us-sandbox-courier-<*>.push-apple.com.akadns.net.\t300 IN A <*>"", ""us-sandbox-courier-<*>.push-apple.com.akadns.net.\t300 IN A <*>"" ] }",1
01b88478,"{ ""handling_time"": <*>, ""client"": ""<*>"", ""client_port"": <*>, ""qname"": ""<*>-courier.sandbox.push.apple.com"", ""view"": ""policy_view_<*>"", ""type"": ""AAAA"", ""result"": ""OK"", ""ret"": [ ""<*>-courier.sandbox.push.apple.com. <*>\tIN CNAME <*>.courier-sandbox-push-apple.com.akadns.net."", ""<*>.courier-sandbox-push-apple.com.akadns.net. <*> IN CNAME us-sandbox-courier-<*>.push-apple.com.akadns.net."" ] }",1
```

  - 可以看到此时部分Json内容直接被判断为模板

算法实现（三）

- 重分集群
  - 计算日志之间的欧式距离，按照相似度再进行一次集群划分：

$$S_q = \frac{\sum_{i=1}^{\max(n_1,n_2)} E(S_1(i), S_2(i)) - NE(S_1(i), S_2(i))}{\max(n_1, n_2)}$$

  - E和NE分别代表相同的两个词和不同的两个词
  - 如：
  - client response: { " handling_time" : "*" }
  - client response: { " handling_time" : "<*>" }
  - 现在将会被分到同一集群

算法实现（四）

- 模板提取
  - 计算模板（C）每个单词（w）在所有模板（S）中出现的概率：

$$P(w) = \frac{|G\{w \mid w \in C\}|}{|S|}$$

  - 在原有的简单共有词相似度计算中，引入带系数的差异度：

$$S_{new} = \frac{|X| - a(Max(|A|,|B|) - |X|)}{Max(|A|,|B|)}$$

算法实现（五）

- 模板提取
  - 设计BMerge算法，使用算法将相似度（S）大于阈值（T）的模板进行合并
  - 不同的日志有时需要采用不同的阈值来进行处理

BMerge 算法

输入： W_1, W_2

输出： 无

```
1: function BMerge(W1, W2)
2:     for all w1 ∈ W1, w2 ∈ W2 do
3:         if w1 = w2 & P(wi) < P(wi−1) then
4:             RemoveFromBoth(w1, w2, W1, W2)
5:         else if w1 ≠ w2 then
6:             RemoveFromLong(w1, w2, W1, W2)
7:         end if
8:     end for
9: end function
```

三、实验过程

实验数据
实验结果

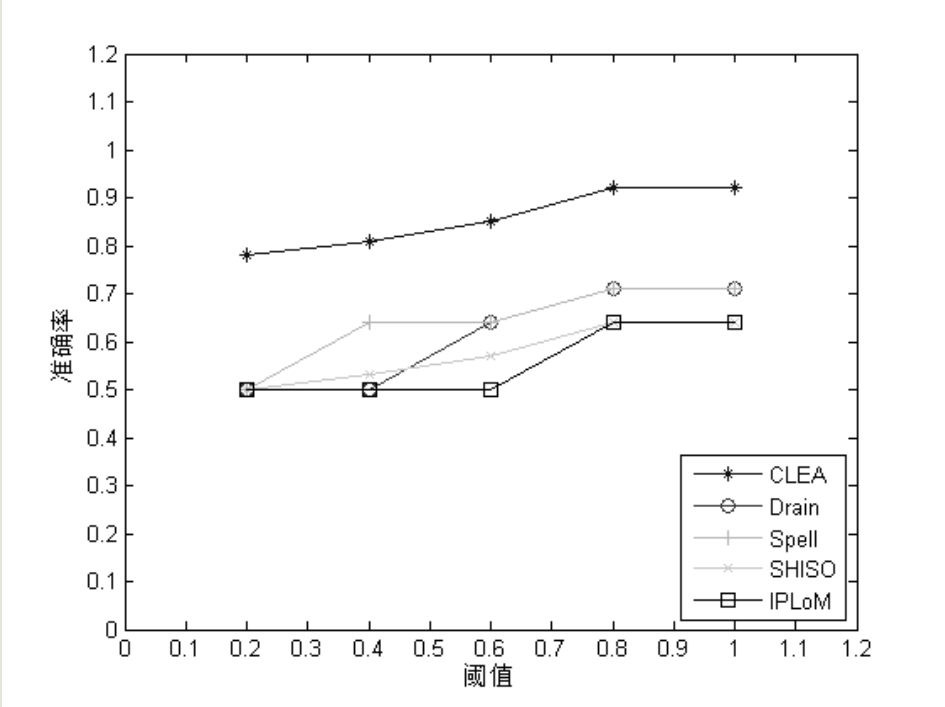
实验数据

来源：北京邮电大学07.03—07.04期间的DNS日志与交换机日志

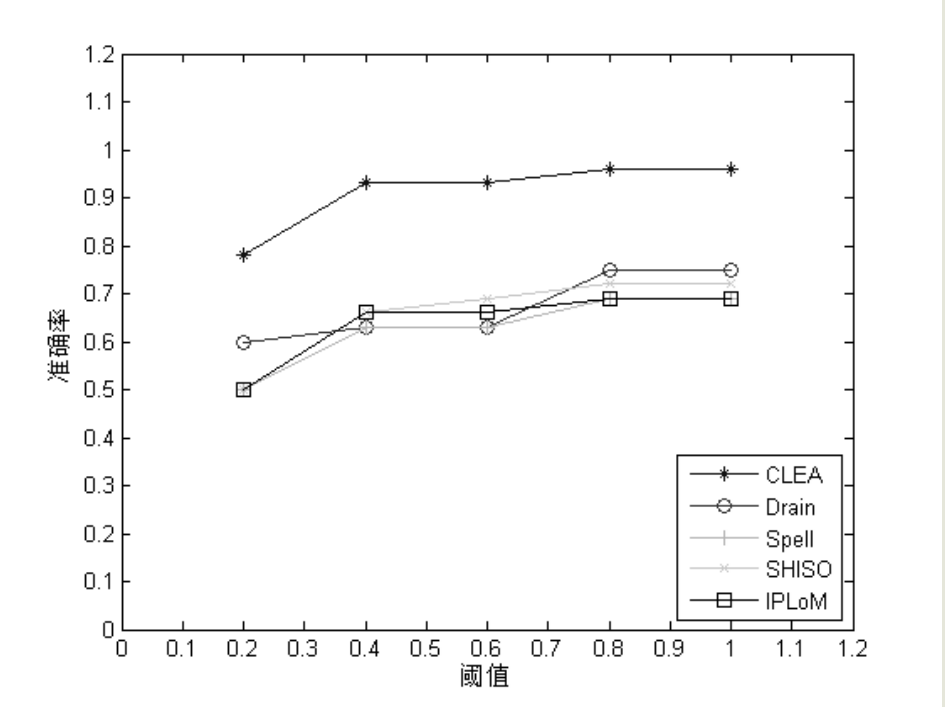
- DNS日志
  - 体积约为10G左右，其中包含了Json字段
- 交换机日志
  - 体积约为7G左右，其中包含了Key-Value的字段

实验结果（一）

DNS日志解析准确度



交换机日志解析准确度



实验结果（二）

- Drain算法结果：
  - { "handling_time": "*", "client": "*", "view": "policy_view_*", "type":"A", "result": "OK", "ret": [* * * *] }
  - { "handling_time": "*", "client": "*", "view": "policy_*", "type":"AAAA", "result": "OK", "ret": [* * * * * *] }
  - { "handling_time": "*", "client": "*", "view": "policy_view_*", "type":"AAAA", "result": "NXDOMAIN" }
  - { "handling_time": "*", "client": "*", "view": "policy_*", "type":"NS", "result": "NXDOMAIN" }
- 使用复合型日志模板解析之后：
  - { "handling_time": "*", "client": "*", "view": "*", "type":"*", "result": "OK", "ret": [*] }
  - { "handling_time": "*", "client": "*", "view": "*", "type":"*", "result": "NXDOMAIN" }

谢谢聆听！