Improvement of Log Pattern Extracting Algorithm Using Text Similarity

ZHAO Yining
Computer Network Information Center, Chinese Academy of Sciences
in HPBDC18, 2018/05/21
Content

- CNGrid & LARGE
- Why Log Patterns & Extracting Algorithm
- Algorithm of Identical Word Rate
- Text Similarity Based Approach
  - Improved Extracting Formation & LCS
  - Experiment Result
- Modified Log Comparing Model
- Summary & Future Work
CNGrid & LARGE

- China National HPC Environment

2 Operating Centers (Beijing / Hefei)

19 Sites (200PF + 162PB)

Portal with Micro-Service Architecture

Application oriented Global Scheduling & Predicting

Resource Evaluation Standard & Comprehensive Evaluation Index
CNGrid & LARGE

- Log Analyzing Framework in Grid Environment
We want to be alerted for logs in certain patterns, but...
- too many logs for human to read
- need to summarize patterns before defining alert rules

Set of log patterns in our context:
- patterns are different from each other
- covering all logs in original set
- significantly less than original

The process of using log patterns
- filter and remove frequent normal logs
- use log pattern extraction algorithms to get the set of patterns
- manually check the set and pick out abnormal patterns
- define rules to generate alerts for these patterns
Algorithm of Identical Word Rate

- Algorithm of identical word rate – a straightforward way
  - identical words
    - 2 words that are identical
    - and in the same position in 2 original logs
  - identical word rate
    - (number of identical words) / (total words)
    - predefined threshold $t$
    - If IWR is greater than $t$, the two logs are in one pattern

- Process of algorithm of IWR
  - set threshold $t$ and initial empty pattern set $P$
  - for each new incoming logs, compute IWR with each pattern in $P$
  - if pattern matched, skip to next; if none matched, add to $P$

- Significant Limitation
  - Logs with different length has IWR of ZERO!
Using Text Similarity to resolve the problem

- \( S = P \times O \)
- \( S: \) similarity, \( P: \) proportion of common words, \( O: \) order factor

Two logs \( l_1 \) and \( l_2, L_1 \) and \( L_2 \) are word sets respectively

- define \( P: P(l_1, l_2) = (|L_1 \cap L_2| \times 2) / (|L_1| + |L_2|) \)
- define \( O: O(l_1, l_2) = \text{SeqSim}(l_1, l_2) / |L_1 \cap L_2| \)
- hence \( S: S(l_1, l_2) = (\text{SeqSim}(l_1, l_2) \times 2) / (|L_1| + |L_2|) \)

By this, logs in different lengths can be compared
Text Similarity Based Approach (2)

- Using Longest Common Subsequence to define $\text{SeqSim}(l_1, l_2)$
  \[ S(l_1, l_2) = \left( \frac{|LCS(l_1, l_2)| \times 2}{|L_1| + |L_2|} \right) \]
  - Same pattern if $S(l_1, l_2) \geq t$, where $t$ is the predefined threshold

- The process of improved log pattern extracting algorithm
  - set the threshold value $t$. Set the initial log pattern set $P$ to be an empty set
  - for a new log $l$ appearing from the input log set $L$, compute $S_i(l, p_i)$ between $l$ and every $p_i \in P$ using a LCS algorithm
  - if there is no $S_i(l, p_i) \geq t$, add $l$ to $P$
  - after all logs in $L$ have been checked, return $P$

- Increase time cost for single comparison
  - but reduce total number of comparisons
  - can be offset by choosing a better LCS algorithm
Text Similarity Based Approach (3)

- Experiment result
  - numbers of extracted patterns

![Graphs showing comparison between IWR and LCS for 'cron', 'maillog', 'secure', and 'message'.]
Text Similarity Based Approach (3)

- Experiment result
  - time costs of candidate algorithms (in milliseconds)

![Graphs showing time costs for different algorithms in cron, maillog, secure, and messages datasets.](image-url)
The original model is bad in time cost of searching patterns
- has to visit all patterns until the one is met

Use hashmap to accelerate the matching
- divide pattern set into subsets by initial words
- skip majority of patterns in irrelevant subsets

Matching process:
1. get initial word of the log
2. hash the word
3. find desired subset in hashmap
4. compare with patterns in the subset
This approach cannot deal with patterns with unfixed initials
- build an unfixed pattern set

In real system, we split pattern set in 4 parts:
- fixed alert pattern set
- unfixed alert pattern set
- fixed normal pattern set
- unfixed normal pattern set

When a new log comes, it is compared in the 4 sets in turn to decide processing methods
Real time cost comparison between original & modified models
Summary & Future Work

- Log patterns: used to build log recognition
- Algorithm of IWR isn’t capable to match logs in different lengths
- Using the idea of text similarity and LCS to improve the algorithm
- Modify log comparing model to accelerate the process

Future work: log pattern based analyses in CNGrid
- log pattern associations
- log flow feature modeling