Parallel and Distributed Pattern Mining

Ishak H.A Meddah, Labri Laboratory, École Supérieure en Informatique, Sidi Bel Abbès, Algeria
https://orcid.org/0000-0003-4164-7487

Nour El Houda REMIL, Mascara University “Mustapha STAMBOULI”, Mascara, Algeria

ABSTRACT

The treatment of large data is difficult and it looks like the arrival of the framework MapReduce is a solution of this problem. This framework can be used to analyze and process vast amounts of data. This happens by distributing the computational work across a cluster of virtual servers running in a cloud or a large set of machines. Process mining provides an important bridge between data mining and business process analysis. Its techniques allow for extracting information from event logs. Generally, there are two steps in process mining, correlation definition or discovery and the inference or composition. First of all, their work mines small patterns from log traces. Those patterns are the representation of the traces execution from a log file of a business process. In this step, the authors use existing techniques. The patterns are represented by finite state automaton or their regular expression; and the final model is the combination of only two types of different patterns whom are represented by the regular expressions \((ab)^*\) and \((ab^*c)^*\). Second, they compute these patterns in parallel, and then combine those small patterns using the Hadoop framework. They have two steps; the first is the Map Step through which they mine patterns from execution traces, and the second one is the combination of these small patterns as a reduce step. The results show that their approach is scalable, general and precise. It minimizes the execution time by the use of the Hadoop framework.

KEYWORDS

Hadoop, MapReduce, Process, Trace

1. INTRODUCTION

Many techniques have been proposed that mine such patterns from execution traces. However, the most existing techniques mine only simple patterns, or a single complex pattern that is restricted to a particular set of manually selected events.

Recent work has recognized that patterns can be specified as regular languages (Ammons, Bodík, & Larus, 2002). This allows the compact representation of patterns as regular expressions or finite state automata, and it allows the characterization of the pattern mining as a language learning problem.

Current approaches are fundamentally similar; each takes as input a static program or a dynamic traces or profile and produces one or more compact regular languages that specify the pattern representation or the workflow. However, the individual solutions differ in key ways.

In this paper, we present a new general approach to patterns mining that addresses several of the limitations of current techniques. Our insight is twofold. First, we recognize that instances of smaller patterns which can be composed in parallel into larger patterns. Second, we also observe that the composition of small pattern can be in parallel.

DOI: 10.4018/IJRSDA.2019070101

Copyright © 2019, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.
Then, we leverage this insight to divide our work into two parts; The first one, we use a technique on how we can mine two types of small patterns and we compose them by using standard algorithms for finite state automaton manipulation, and some special rules used by M. Gabel and Z. Su (2008), The mining is also performed by symbolic mining algorithm (Gabel & Su, 2008; Zhang, Zhu, Chen, & Yang, 2015).

For the second one, we use the framework MapReduce in mining and composing micro-patterns. Those patterns have been shown as regular expressions or their finite state automata. In this part we mine small patterns using the same symbolic mining algorithm but in parallel as Map step, and we compute this small pattern into a larger pattern in parallel as a reduce step.

Our work is an amelioration to the existing techniques in mining patterns and parallel distributed process.

Our approach has been implemented in the java programming language with two log files of two applications namely: SKYPE and VIBER. The size of the first log file is 10 GB, and the second is 18 GB, which are generated by log file generator.

We have tested our approach in three clusters in a cloud, the first regroups five machines, the second regroups ten machines, and the third regroups 20 machines. The traces in our applications are the call, the answer, and the messages, etc.

2. RELATED WORK

Many techniques are suggested in the domain of process mining; we quote:

M. Gabel et al. (2008) present a new general technique for mining temporal specification, they realized their work in two steps, firstly they discovered the simple patterns using existing techniques, then combine these patterns using the composition and some rules like Branching and Sequencing rules.

Temporal specification expresses formal correctness requirement of an application’s ordering of specific actions and events during execution, they discovered patterns from traces of execution or program source code; The simples patterns are represented using regular expression (ab)* or (ab*c)* and their representation using finite state automata, after they combine simple patterns to construct a temporal specification using a finite state automaton.

G.Greco et al. (2006) discovered several clusters by using a clustering technique, and then they calculate the pattern from each cluster, they combine these patterns to construct a final model, they discovered a workflow scheme from, and then they mine a workflow using a mine workflow algorithm, after they define many clusters from a log traces by using clustering technique and process discover algorithm and some rules cluster.

Then they use a find features algorithm to find patterns of each cluster, finally they combine these patterns to construct a completely hierarchical workflow model.

In their clustering algorithm, clusters reflect only structural similarities among traces; they say that in future works extending their techniques to take care of the environment so that clusters may reflect not only structural similarities among traces, but also information about, e.g., users and data values.

H.R. Motahari-Nezhed et al. (2008) use a service conversation log: first they split a log into several partitions, second they discovered a model from each partition, and third, they annotate the discover protocol model with various metadata to construct a protocol model from real-word service conversation logs.

The protocol is the specification of all possible conversations that a service can have with its partners and the conversation consists of a sequence of messages exchanged between two or more services.

During the split they discovered a simple precise protocol models by analyzing messages sequences in the log, they eliminate conversations considered noisy or not presented in the log; they
Evaluating the Degree of Trust Under Context Sensitive Relational Database Hierarchy Using Hybrid Intelligent Approach
