Chapter 20
Efficient Implementation of Hadoop MapReduce-Based Dataflow

Ishak H. A. Meddah
Oran University of Science and Technology – Mohamed Boudiaf, Algeria

Khaled Belkadi
Oran University of Science and Technology – Mohamed Boudiaf, Algeria

ABSTRACT

MapReduce is a solution for the treatment of large data. With it we can analyze and process data. It does this by distributing the computation in a large set of machines. Process mining provides an important bridge between data mining and business process analysis. This technique allows for the extraction of information from event logs. Firstly, the chapter mines small patterns from log traces. Those patterns are the representation of the traces execution from a business process. The authors use existing techniques; the patterns are represented by finite state automaton; the final model is the combination of only two types of patterns that are represented by the regular expressions. Secondly, the authors compute these patterns in parallel, and then combine those patterns using MapReduce. They have two parties. The first is the Map Step. The authors mine patterns from execution traces. The second is the combination of these small patterns as reduce step. The results are promising; they show that the approach is scalable, general, and precise. It minimizes the execution time by the use of MapReduce.

INTRODUCTION

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

Recent work has recognized that patterns can be specified as regular languages (Ammons et al., 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.

DOI: 10.4018/978-1-5225-3004-6.ch020

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 pattern mining that addresses several of the limitations of current techniques. Our insight is twofold. First, we recognize that instances of smaller patterns can be composed in parallel into larger patterns. Second, we observed also that the composition of small pattern can be in parallel.

We then leverage this insight to divide our work into two parts; The first one, we use a technique how we can mine two types of small patterns and we compose them by using standard algorithms for finite state automaton manipulation (Gabel & Su, 2008), and some special rules using by M. Gabel and Z. Su (2008), the mining is also performed by symbolic mining algorithm (Gabel & Su, 2008, May).

The second one, we use the framework MapReduce in mining and composing micropatterns; those patterns have been shown as regular expressions or their finite state automatons, in this party we mine small patterns using the same symbolic mining algorithm but in parallel as Map step, and we compute these small patterns into larger pattern in parallel as reduce step.

Our approach has been implemented in the java programming language with the log file of two application; the SKYPE and VIBER applications. The size of those applications log file is more than 10 Go, who are generated by log file generator.

We have tested our approach in two clusters in a cloud, the first regroup five machines, and the second regroups ten machines, the traces in our applications are the call, the answer, and the messages.

RELATED WORK

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

M. Gabel and al (Gabel & Su, 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 automaton, after they combine simple patterns to construct a temporal specification using a finite state automaton.

G.Greco and al (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 a 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.