1. Introduction

1.1. Background

Structural changes are often seen in time series data. This observable behavior is highly appealing to statistical modelers who want to develop a model which is well explained. A method to detect changes in time series data when a time instant is unknown is called change point analysis (Basseville et al., 1993). It has a bottom line for discovering the time point where the changes in time series occur. Change point analysis can be referred to different kinds of name such as breakpoint and turning point. However, *change-point* is the commonly used term when the point in time series takes place. Another important term which will be used in this thesis is *regime switch* which refers to persistent changes in time series structure after the occurrence of change point (Weskamp and Hochstotter, 2010). Change point analysis has been studied over decades as it is a problem of interest in many applications which the characteristic of data is collected over time. For instance, it can be utilized in medical condition monitoring (Staudacher et al., 2005)in order to evaluate the sleep quality of patients based on their heart rate condition. Another example is the application in climate analysis (Reeves et al., 2007; Beaulieu et al., 2012), where temperature or climate variations is detected. This method gradually becomes important over the past few decades due to the effects of the global warming and the increases in greenhouse gas emissions. Change point analysis can also be found in the field of quality control (Page, 1954) in industrial production. Since it is a continuous production process, in mass production process, if the product controlled value is not monitored and exceed the tolerant undetected, it could lead to the lost of a whole production lot. Other applications of change point analysis are identifying fraud transaction (Bolton and Hand, 2002), detecting anomalies in the market price (Gu et al., 2013) and detecting signal processing (Basseville et al., 1993) in streaming data as well. The change should be flagged as soon as it occurs in order to be properly dealt with such changes in time and reduced any possible consequences (Sharkey and Killick, 2014).

In this study, change point analysis will be used to identify changes in performance of Ericsson's software products. Many test cases have been executed for testing software packages in a simulation environment. Before launching the software products to its customers, the company needs to test and determine how is software package performs. The performance of these software packages is evaluated by considering on CPU utilizations (percentages of CPU's cycle spent on each process), memory usage and latency.

Recently, a method called hidden Markov model or Markov switching model is widely used for discovering a change point in time series. Both terms are acceptable but being called differently regarding to the different fields of study. Markov switching model uses a concept of Markov process to model an underlying segmentation as different states and then specify a distinct change of location. Hence, the method is able to identify a switch in time series when change point occurs (Luong et al., 2012). This method is generally used in almost all current systems in speech recognition (Rabiner, 1989) and found to be important in climatology such as describing the state in the wind speed time series (Ailliot and Monbet, 2012) and in biology (Stanke and Waack, 2003) where the prediction of gene is being made. Markov switching model has been extensively applied in the field of economics and finance and has a large literature. For example, business cycle can be seen as hidden states with seasonal changes. The growth rate of gross domestic product (GDP) is modeled as a switching process to uncover business cycle phases i.e., expansion and recession. The fitting model can also be used to understand the process where there is a transition between the economic state and the duration of each period (Hamilton, 1989). In finance data, time series of returns is modeled in order to investigate stock market situation i.e., bull or bear market (Kim et al., 1998).

Markov switching model is one of the most well-known non linear time series models. This method can be implemented to model various time series data with dynamic behavior. The structural changes or regime shifts in data mean that constant parameter in time series model might be insufficient to capture these behaviors and describe their evolution. Markov switching model takes the presence of shifting regime in time series into account and models multiple structures that can explain these characteristics in different states at different time. The shift between state or regime comes from the switching mechanism which is assumed to follow an unobserved Markov chain. Thus, the model is able to capture more complex dynamic patterns, identify the change of location and regime switch in time series.

Each software package in the system is viewed as a time point in time series and the performance of software package is treated as an observe value. According to the behavior of observation sequence in this study, it is found that observation is not completely independent of each other (i.e., performance of current software package depends on the performance from the past version of software package). Therefore, additional dependencies at observation level with the first order autoregression is taken into consideration when modeling the Markov switching model. It is simply called Markov switching autoregressive model.

1.2. Objective

The main idea of this thesis is to reduce laborer work to do visual inspection whenever they want to analyze the performance of update software package. With the rise of data generation coming from a large number of test cases, this work becomes much more difficult and perhaps inefficient to do it manually. The main objective for this thesis is to implement machine learning, the algorithm that has an ability to learn from data, to analyze the performance of the software package. The algorithm will help indicate whether the performance of software package is in a degradation, improvement or steady state. There is also a case when changes in the test environment affect performance even though there is no change in the software package. The implemented algorithm should be able to also detect when the test environment is altered. CPU utilization is focused in this thesis. It is one of the essential factor that needs to be optimized when considering releasing an upgrade software package.

To summarize, this thesis aims to:

- Detect the degradation, improvement or steady state in CPU utilization
- Detect whether there is some changes in test environment that have an impact on CPU utilization

The thesis is structured as follows: Chapter 2 provides detail and description of datasets used in the analysis. Chapter 3 presents methodology. Results from the analysis is shown in Chapter 4 along with tables and plots. Chapter 5 discusses about the outcome and the obtained results, and conclusion can be found in Chapter 6.