1. Introduction

1.1. Background

Change point analysis is a method to detect changes in time series data when time instant is unknown (Basseville et al., 1993) and has a bottom line for discovering and estimating the time point where the changes in time series occur. Serveral terms such as breakpoint or turning point are also used to denote the respective events. However, the commonly used term when the point in time series takes place is *change-point* and the term *regime switch* refers to the different of regimes after the occurence of change point (Weskamp and Hochstotter, 2010). It has been studied over decades as it is a problem of interest in many applications in which the characteristic of data is collected over time. Change point analysis can be utilized in medical condition monitoring (Staudacher et al., 2005), to evaluate the sleep of patients based on their hearth rate variability. It is applied in crimate analysis (Reeves et al., 2007; Beaulieu et al., 2012), where temperature or climate variations is detected. It gradually becomes important over the past few decades due to the effects of global warming and a huge growth in the greenhouse gas emissions. Applications can be found in the field of quality control (Page, 1954)in which the quality of industrial product is of interest. Since it is a continuous production process, if something goes wrong at some point of the process (i.e., defective products, machine breakdowns), it can lead to a large amount of quality worsens. Method is also used for identifying fraud transaction once it is commited (Bolton and Hand, 2002), and detecting anomalies in the market price (Gu et al., 2013). Signal processing also benefits from the change point analysis by detecting significant changes in the streaming data (Basseville et al., 1993). The change should be flagged as soon as it occurs in order to be properly dealt with such changes in time and reduced any consequences that could have happened (Sharkey and Killick, 2014).

In this study, change point analysis will be used to identify the changes in performance of Ericsson's software product. Many test runs have been executed for testing the software packages in the simulation environment. Before launching product to customer, company needs to test and determine how a software package performs in general. The performance of these software packages are evaluated by considering on the CPU utilization, a percent of the CPU's cycle that spent on each process, and some other performance metrics (e.g., memory usage, latency).

Recently, method which is now becoming more popular for addressing such changes in time series is a hidden Markov model. It uses a concept of Markov process where the system treats data as observations and tries to model an underlying segmentation as states. Hence, it is able to identify the switch in hidden states when change-point is most likely to occur (Luong et al., 2012). A hidden Markov model is widely 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. It has been extensively applied in the field of economics and finance and has a large literature. For instance, business cycle can be seen as hidden states with seasonal changes. It is modelled as a switching process to uncover the state over expansions and recessions. Model can also be used to understand the transition between the economic state and the duration of each period (Hamilton, 1994). Financial time series is modelled in order to investigate how stock market is doing in general i.e., bull or bear market (Kim et al., 1998). Markov regime switching model is what econometrician and people who study related to this field always address to when refer to the hidden Markov model.

The term Markov regime switching model will be used throughout the thesis. Markov regime switching model is one of the most well-known non linear time series models. It takes the behavior of shifting regime in time series into account and models multiple structures that can explain this characteristic 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 complex dynamic patterns, identify the change of location and regime switch in time series.

Each software package in the testing system is viewd as a time point in time series and the performance of software package is 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 evaluated from the past version of software package). Therefore, additional dependencies at observation level with the first order autoregression is taken into consideration when modelling the Markov regime swiching model. It is simply called Markov switching autoregressive model.

1.2. Objective

The main idea of this thesis is to reduce the 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 runs, 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 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.