5. Discussion

In this chapter, a discussion of the model selection for each given dataset is explained. Then, a state inference from the model is made. Lastly, a results discussion of this study is provided.

5.1. Model selection

In this analysis, a three-state model which had all switching coefficients from Analysis I (see sec. 4.1) acted as a baseline model for each dataset. This model was used to compare with other models which had different combination of switching coefficients. If the compared model has higher BIC than the baseline model, its performance is concluded to be inferior and should not be considered for any further analysis. However, only examining the BIC when choosing a model for the data is insufficient. Remark that BIC might be unreliable for a small or moderate dataset as BIC is derived under the assumption of a normal distributed data (Rydén et al., 2008). Therefore, other aspects should also be taken into account along with the BIC such as model outputs and plots.

Software release L16A Even though model 3 had the lowest BIC, a coefficient of *Paging* in one state had a zero standard error which led t-value to infinity. This zero value can be interpreted as an actual zero or an extremely small value that a computer treated as zero because significant digit is lost. Nevertheless, either way suggests that this model is not a good model to be used with this dataset as the model might be overfitting with the training data. The standard error of zero means that there is no variation in the data i.e., every data value is equal to the mean value. Therefore, model 1 which had the second lowest BIC was chosen for this given dataset instead.

Software release L16B Models 2 and 6 had the lowest BIC among the other models. Nonetheless, their plots are similar to each other and provided a difficulty in interpreting results (see Figure B.1). Observations stay in State3 in the first half of the period but stay in State2 in the second half of the time period. There are continuous fluctuations of the CPU utilization value through out the whole time period. Therefore, for observations (or software packages) to remain in the same

state for a long duration without switching to other states seem unrealistic. The selected model for this dataset was model 4 where its BIC was in the third lowest place. Even though model 4 had slightly higher BIC than models 2 and 6, the model produced more sensible result.

Software release L17A The first four models with lowest BIC (Models 1, 5, 3, and 7, respectively) had similar results both in model outputs and plots. Thus, Model 1 which had the lowest BIC was chosen for this dataset.

5.2. State inference

Another important task after the suitable model was decided is to make an inference on the derived states because the model output does not provide any definition of these states. Therefore, an interpretation and inference of the state need to be specified. The state inferences for each dataset are shown below.

Software release L16A (see Figure 4.7) Despite periods where there is a slightly decrease in CPU utilization value, State1 contains two peaks which have the value of the CPU utilization higher than 300. This clearly implies that test cases in State1 perform badly and should be defined as a *Degradation* state. As for State2 and State3, both states can be viewed as an Improvement state. However, State2 seems to have many periods where test cases drop to a lower value in the CPU utilization. State3 also has two periods of abrupt changes to higher CPU utilization values. Therefore, State2 appears to be an *Improvement* state while State3 is said to be a *Steady* state.

Software release L16B (see Figure 4.8) There are many high peaks occurred in State1, and test cases tend to increase the values of the CPU utilization when they enter this state. Hence, State1 is said to be a *Degradation* state. State2 could be characterized as an *Improvement* state. The reason is that a decreasing pattern of the CPU utilization value for the test cases in State2 can be seen in most of the period. State3 contains the period of positive and negative peaks, and does not appear to have a specific characteristic for the state of the CPU utilization. The difficulty of making an interpretation for State3 infers that the state may be defined as a *Steady* state.

Software release L17A (see Figure 4.9) Abrupt changes in the CPU utilization values when test cases enter State1 could be seen in most of the period. Despite low values in some of the test cases, State1 appeared to have more high value in the CPU utilization. Hence, State1 could be defined as a *Degradation* state. The CPU

utilization values clearly decrease when test cases enter State2. These test cases often switch from State1, which is thought to be the state where the performance of the test cases perform badly. As a consequence, this behavior indicates an improving in the test cases. Thus, State2 is defined as an *Improvement* state. The test cases which stay in State3 have an unclear pattern in the CPU utilization. The values are quite fluctuate and because of this behavior, the state is labeled as a *Steady* state. The only peak in the time series data belonged to State3, and will be viewed as an anomaly in this case.

5.3. Results discussion

A thorough search of relevant literature yielded that this thesis work might be the first time that Markov switching model has been applied to the data of CPU utilization. In previous works, this model was mainly applied with financial data or signal processing which were used as an inspiration for this study. However, because of differences in the characteristic of data, the procedure was slightly adjusted.

In this study, *NodeName*, which is used to execute test cases, was assumed to be indifferent i.e., performance of test cases are the same regardless of the machine. Therefore, selecting a minimum value of the CPU utilization of the test case for each software package in data preprocessing step is reasonable.

There is no fixed rules in deciding the number of states. Therefore, difference number of states were tried with the method in order to determine which number would be best for this analysis. The results from the model with two states offered less details and did not cover all happening situations. Furthermore, all the graphs of the two-state model are unrealistic and also difficult in interpret. Despite higher BICs for the dataset of the software release L16B and L17A, three-state model provides more interpretable plots and better fit to the time series data. The four-state Markov switching model for each software release was also briefly tested in the analysis. Apparently, defining four states caused more difficulty when fitting the model and making an inference on the states. The results were rather poor when compare to the model with two and three states and was not include in the report. Higher number of states are more likely to give worse results and were not considered. Thus, the number of chosen states after applying Markov switching model to each dataset was three.

Among all the predictor variables, the local events variables in *EventPerSec* is more essential than the test environment variables. Having a constant value for the local events variables would be inadequate to characterize the whole time series data and its evolution. However, the test environment variables do not necessarily require a switching mechanism. This is why in this study, the test environment variable was possible to not have a switching effect.

Another thing worth mentioning is that when modeling the Markov switching model,

it is better to try estimating a simpler model first. The size of the model grows exponentially according to the number of states k, and the number of predictor variables n. In addition, if all coefficients have switching effects, the model will have more parameters to be estimated. The function in the package will try to fit the model and estimate the value of the coefficients, but there is no guarantee that the obtained results of the likelihood will be a global maximum. As a consequence, the estimated values of the coefficients cannot be completely trusted. Perlin (2015) suggested any model with $k > 3$ and $n \geq 4$ is not recommended. This is also the main reason why not all the local events were included, and the model was not made to have all the switching coefficients.

As mentioned previously in sec. 4.1, a variable called *DuProdName* in the dataset of the software release L16A was excluded from the regression model as it caused a singularity problem. This problem could be reduced by increasing the size of the data. However, with a limited available data, fifty-seven observations in this dataset, the variable *DuProdName* has to be dropped from the regression model before doing a further analysis.

For each software release, each state in the model has a significantly high r-squared value which is greater than 0.9 (see Appendix B). R-squared value is an intuitive measurement to determine how well the model fits with the data. In general, the higher r-squared value is more preferable. Nevertheless, high r-squared value does not necessarily indicate that the model is a good model because the model with high r-squared value could be overfitting due to too many predictor variables were used in the model. Using only r-squared value to determine adequacy of the model can be misleading. A residual analysis should be considered when evaluating a model.

A Q-Q plot is a visual check for an assumption of normality. The plot provides a rough idea whether the assumption is plausible or not. From the residual analysis in sec. 4.3, it revealed that the residuals of all three datasets did not completely follow the normal distribution. This is not surprising as real data rarely have a well-defined structure. Since the Markov switching model assumed normal distributed residuals, the chosen models might have less credible results. In addition, the dataset used in this thesis is not sufficiently large. As a result, the Q-Q plot is often unclear and difficult to spot its basic feature due to more noises. ACF and PACF plots are commonly used to check a randomness in a data and also to identify a possible structure of a time series data. The models of all three datasets seemed to capture the dependent structure of the error terms in the data, even though a small amount of autocorrelation were left in the residuals. Moreover, these plots indicate that an AR(1) is already justified for these datasets.

More importantly, the results of the Markov switching model could be affected by various types of bias. Due to a small size of the training data, the training model will not be very effective and unable to capture the behaviors of new observations that lie outside the range of the training data. This could be seen in a state prediction of the software release L16A (see sec. 4.6). Secondly, only the predictor variables that

have been analyzed to have significant impacts on the CPU utilization were included in the model. However, many other variables that is possible to have minor impact on the CPU utilization but were overlooked. Therefore, there might be some bias by not including these variables into the analysis. Finally, the criteria that has been used to select the number of states and the number of switching coefficients in the model are rather subjective. Hence, these chosen number might not be the best for different perspectives and so could be counted as one of the bias as well.

A benefit of using a non-parametric analysis is that no assumption of the data distribution is required, and all type of distributional change within a time series can be detected. However, the E-divisive method was proved to have less power in detecting changes in the data. The main reason that the E-divisive method is not as powerful as the Markov switching model is because the Markov switching model included variables that have influences on the CPU utilization into the analysis but the E-divisive method only considered the values of the CPU utilization.

One behavior found from the Figure 4.13 and Figure 4.14 is that if a pattern of a state in the data is not obvious, the E-divisive method will be unable to detect locations of the change points. For instance, the method could not identify any changes in the first hundred observations, and also at the end of the time series in the simulated Dataset 1. This is more clear when examining the results in the simulated Dataset 2 where the E-divisive method could only detect two change point locations. If the response variable value fluctuates but the average value does not change drastically, the E-divisive method will not be able to identify the state change.

Despite some false alarms and missed detections as seen when the methods were tested with the simulated datasets, the detections are prone to be accurate when both the Markov switching model and the E-divisive method indicate the change at the exactly same location. When applying the methods to the real data, it could be concluded that the state change actually happens if both methods locate the change at the exact location. However, the location of the occurrence is possible to be slightly missed located. Furthermore, if both methods could detect the change at the close but not exact location, there are small chances that the detections could be false alarms.

In sec. 4.7, the model evaluation for each model was performed. The Markov switching model of the simulated Dataset 1 was able to predict the state for the test set considerably well. Estimated coefficients in each state were similar to the coefficients in actual models as well, except for an intercept of State2. State2 was considered to be a *Normal* state. The deviation between the actual and estimated coefficient of the intercept in this state is probably a reason why the model predicted observations from other states as the *Normal* state. For the simulated Dataset 2, estimated coefficients from the model were not close to the actual coefficients, especially the values of intercept in all states. State2 or said to be a *Good* state had totally different value between the estimated and actual coefficient. Therefore, the model had very

worse performance when predicting observations that came from the *Good* state. However, the overall accuracy of the model was still rather high indicated that the model performed well in identifying the switches between states.

From the results obtained in this section, it can also be implied that a state prediction function, an additional implemented function in the package, is able to work properly.