ECONOMICS & INFORMATICS
ERASMUS SCHOOL OF ECONOMICS
ERASMUS UNIVERSITY ROTTERDAM

MASTER THESIS

A State-Based Prediction Model for Energy Demand Forecasting

Supervisor:
Dr. Yingqian Zhang
Co-supervisor:
Dr. Wolfgang Ketter

Author:
Kevin Meijer
312177

24th of May 2012

Abstract

In this thesis we introduce the state-based prediction method that is applied on the domain of Power TAC for performing short-term energy demand forecasts. Power TAC is an agent-based competition that simulates an energy market in which energy broker agents compete against each other for the goal of profit maximization. One of the tasks of a Power TAC broker agent is to predict the short-term imbalance of energy supply and demand in its portfolio. The state-based method is designed to perform this kind of forecasting. Its main feature is that it uses states that each represents unique combinations of data features to acquire data relevant for predictions. Subsequently, a weighted method as well as a simple linear regression model is used to determine future energy imbalance. The state-based method is compared to the CART regression tree model in terms of prediction performance and time performance. We found by conducting experiments that overall the state-based method obtains better prediction performance, but that it is less robust to noise. In terms of time performance, the state-based method shows a considerable improvement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and background

Agent-based competitions are contests in which designed intelligent agents compete against each other in a simulated environment. Franklin described agents as ‘a system situated within and a part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to affect what it senses in the future’ [8]. Within a competition, the agents have a specific goal and have to make decisions to pursuit that goal. In order to reach their goal, agents are composed of decision algorithms. Agent-based environments provide an excellent training ground for building and evaluating different agent decision-making methods. For example, the strategies of several implemented agents have been described and compared for the 2002 Trading Agent Competition [13]. These agent implementations include prediction methods or components such as genetic algorithms [10], neural networks [23], and fuzzy logic [32].

An agent-based competition that presently is active is the Power Trading Agent competition (Power TAC), in which an electric power market is simulated [18]. The aim of this competition is to get insight into how energy brokers behave in the power market under different circumstances. One can sign up for this competition and build an intelligent agent that represents such an energy broker.

The goal of a broker agent is to maximize profits by performing a large variety of tasks. The broker needs to develop a good-quality portfolio consisting of customers who will purchase or sell power. These customers are attracted to the portfolio by setting tariffs. A portfolio is of good quality when it is profitable and balanced. If there appears to be an imbalance between supply and demand of energy, the broker will be penalized with a fee and as a result fails in maximizing its profit. The broker agent thus needs to make its decisions in such a way that the energy imbalance is minimized.

For the version of Power TAC that we use as a basis for our thesis, the version that was used for the Power TAC pilot competition held at International Joint Conference of Artificial Intelligence (IJCAI) 2011, the broker has one option to resolve predicted energy imbalances: to trade in the wholesale market to acquire the deficit or to sell the excess of energy. To be able to know the amounts of energy to trade in this wholesale market, however, first the expected energy imbalance needs to be predicted. I will design and introduce a prediction model that can perform this task. This prediction model will form a component of a Power TAC broker agent framework and will be tested in a simulation environment based on Power TAC. The general setup of this model, however, should also be applicable for other purposes as well, and thus should not only be usable within the framework of Power TAC.

1.2 Questions and methodology

As I will focus on forecasting the supply and demand of power in a Power TAC environment, the research question is stated as follows:

- **How to predict short-term energy imbalance between supply and demand of power of a broker agent in Power TAC?**

To provide a solution to this research question, multiple subquestions are introduced. By dicussing these questions that focus on more specific aspects related to forming predictions in a Power TAC environment, we are able to provide an answer to the research question. The subquestions are as follows:

- What influences the energy consumption and production of customers within Power TAC?
- How to build a prediction model that can accurately predict future energy usage of Power TAC customers?
- How does the applied prediction method hold against another commonly used prediction model?

It is important to know what the Power TAC competition precisely encompasses to get an overview of what aspects should be taken into account for our to-be-built prediction model. Our focus will especially lie on the behaviour of customers within Power TAC. Therefore, we examine the consumption behaviour of customers within the Power TAC pilot competition that was held in July 2011.

In order to build a good prediction model, we will firstly review prediction methods that have been used to perform forecasts. Based on our insights obtained by reviewing other prediction methods and by studying the aspects from the Power TAC pilot competition, we will construct our prediction model known as the state-based prediction method. The prediction performance of this prediction model will be evaluated by forming different scenarios that each are related to specific customer behaviour that may appear in actual Power TAC competitions. These scenarios are constructed using partly data from a default Power TAC simulation and using self-generated data to ensure that these scenarios represent varying customer behaviour.

The acquired prediction performance results will be compared to the CART regression tree prediction approach [6] to examine whether the state-based method has an improved performance over a prediction model that has been used frequently in the past. Since this model shares similarities with our proposed state-based method, it is interesting to find out if the prediction performance of our method is better. Furthermore, we will study and compare the time performance of both prediction methods. In the Power TAC competition, actions need to be executed within a limited time interval [19] and thus it is of vital importance to know whether the methods are capable of performing their predictions in time.

1.3 Thesis outline

This thesis is divided into multiple chapters that each focuses on a different aspect. In Chapter 2 prediction methods and evaluation measures for such methods are reviewed. Chapter 3 provides an overview of Power TAC. In Chapter 4 a model for generating consumption data is defined, and the state-based prediction method is explained in detail. An evaluation of the state-based prediction method is performed in Chapter 5. This evaluation encompasses both prediction performance and time performance in comparison with a regression tree model. In conclusion, Chapter 6 provides a summary of the thesis through a discussion of the previously defined research question and statements related to the subquestions, as well as future work directions for the proposed state-based prediction method.

Additional figures and tables are included in the appendix chapters. Appendix A and B contain data related to the Power TAC pilot competition from July 2011. Evaluation results for both the state-based method as well as the CART regression tree method are summarized in Appendix C.

Chapter 2

Related work

In this chapter we briefly discuss a number of prediction methods and several prediction evaluation measures that have been widely used in the past. Through this discussion, we get an insight into how to build and evaluate a prediction method. In Section 2.1 the prediction methods are described. Then, in Section 2.2 some evaluation measures are explained.

2.1 Prediction methods

There is a wide variety of prediction methods. We will discuss prediction models that can be categorized into four categories. In Section 2.1.1 exponential smoothing methods are reviewed. Section 2.1.2 covers artificial neural network approaches. Then, regression models are discussed in Section 2.1.3. And last, in Section 2.1.4 we describe methods related to decision trees.

2.1.1 Exponential smoothing

What many prediction models have in common is that they use past data to form forecasts. One of the most straight-forward methods is the simple moving average (SMA) method, where a predicted value is equal to the average of a specified number of past values. Similar to the moving average, one can also apply a moving median or geometric moving average. However, due to the high simplicity of these methods, they generally are poor predictors. For example, since all values are weighted equally, the predictions are lagging behind [21]. Furthermore, since no trend is considered the long-term forecasts are a horizontal straight line.

To improve the prediction performance of the SMA method, exponential smoothing methods have been introduced. Varying exponential smoothing methods have been discussed in detail by Gardner [9]. These methods go one step further by assigning different weights to different values. The SMA method considers all past observations equally, while more relevant observations (such as the most recent ones) can be assigned a larger weight to increase their impact of the predicted value. With the simple exponential smoothing method a smoothing factor α is defined to acquire a predicted value. The higher the smoothing factor, the more influence recent values have, and vice versa. Even though simple exponential smoothing methods perform better than SMA methods due to their lower vulnerability to lagging [21], long-term forecasts still suffer from the fact that they are constant because still no trend is taken into account [9].

Double exponential smoothing methods do consider the trend within data and adjust the prediction based on the trend. This trend represents the overall increase or decrease. It is computed in a similar way as the prediction is formed in simple exponential smoothing; a smoothing factor is defined and used as a weight that determines to which degree the previous trend is adjusted by the most recent increase to form the new trend. The larger this factor, the more influence the most recent increase has, and vice versa. The obtained trend then influences the prediction. The larger the trend and the larger the smoothing factor α, the larger the prediction.

In 1960, Winters introduced triple exponential smoothing, which takes into account a seasonal factor next to the trend [31]. In his paper, he illustrated that item sales can greatly vary between different time periods, and thus contain strong seasonal patterns. These time periods with varying data do not necessarily need to represent a season, but could also represent any other time period. By including a season factor as well, the data differences between time periods can be considered in forming predictions.

Additions to the exponential smoothing methods have also been made. Instead of computing a linear trend for the double or triple exponential smoothing methods, models containing exponential trends have also been proposed and are discussed by Pegels [25], Brenner [7], and Roberts [27]. Another modification to exponential smoothing methods is the inclusion of a damped trend. Using a model with a damped trend is preferred over a model without one, since models without damped trends tend to overestimate data beyond the short-term [9] and damped trend models have shown an improvement in prediction accuracy [22, 30].

For double or exponential smoothing methods it is important to properly estimate the applied smoothing factors. Initializing these methods with the wrong parameter will hurt the prediction performance. Finding the right parameter values can be difficult [9].

2.1.2 Artificial Neural Networks

Artificial neural networks (ANN) are prediction models that have been used frequently in the past due to their high predictive performance [29]. They have also been applied for forecasting energy demand [26, 2]. These ANN models are based on the learning process of the human brain, and were first discussed in [23]. The idea of neural networks is that input data is transformed to output data through the use of flexible functions. Neural networks consist of a sequence of layers that each comprises one or more nodes. The output of the nodes from one layer is the input of the nodes of the next layer in the sequence. A neural network consists of one input layer, one output layer, and one or more hidden layers which are between the input and the output layer. To compute the output that results from a certain node given certain input, a weighted function is applied. The sigmoidal function is the most commonly used one in neural networks [29].

In the most basic form of ANN, a feedforward neural network, the data simply flows into one direction through the multiple layers to eventually generate an output value. Normally, however, the weights that are used within the weighted function are constantly updated by using a learning algorithm, which comprises iterative procedures to improve the prediction performance of a neural networks model. The most widely used ANN learning algorithm is the backpropagation model [28]. The backpropagation model encompasses a supervised learning process which compares ANN output values to the target values to optimize the weights that are used in the ANN. In the backpropagation model, we first compute an output value in the same way as for the feedforward neural network. Then the error of the output node is calculated and the weights applied within the hidden layer will be updated with a gradient descent algorithm to minimize the mean squared error. The level to which weights are updated is dependent on the error and a user-defined learning factor. For both the error and the learning factor it holds that the larger the size, the more the weights within the ANN are adjusted. Updating can be applied on a case-to-case interval, which means that weights are updated for each single observation, or done in batches, for which first all observations are run through the ANN before the error is determined and the weights are adjusted [29].

There are some difficulties when using backpropagation neural networks. Even though neural networks have a good predictive performance, their computation time can be larger than that of simpler methods since a neural network must first be trained to set all its parameters. After it is fully trained and a neural network model is ready for use, the problem of large computation time vanishes. However, within Power TAC new data appears almost constantly and thus the neural network needs to be trained regularly throughout a Power TAC game. Within the domain of Power TAC energy consumption forecasting, this can become a problem, as all actions of an agent need to be consistently performed on short intervals.

2.1.3 Regression

Another commonly used method for prediction is regression. Regression methods are one of the most popular methods for making predictions [29]. A straightforward form of regression is simple linear regression. In simple linear regression a linear function is computed between one independent variable and a dependent variable, which respectively serve as the input and output. Even though there are multiple techniques to compute the linear function, the most commonly used method is the ordinary least squares method [24]. With this method, the linear function is determined by minimizing the sum of squared deviations between the predicted values and their target values. This sum is also called the sum of squared residuals. When there is more than one explanatory variable, one can use multiple linear regression, which follows the same mechanics as simple linear regression, but uses at least two explanatory variables instead of one. When the user knows the expected relationship between the independent and dependent variable is close to linear, simple linear regression is a good and simple method to use.

An extension to linear regression is segmented linear regression, in which multiple linear functions are calculated for different intervals of the independent variable that is used [20]. For example, one can compute a linear function for small values and a separate linear function for larger values of the independent variable. This kind of regression is convenient in cases where there are different relationships between the dependent and independent variable given different magnitudes of the independent variable, such as is the case in a parabolic relationship. Simple linear regression would not be able to capture such a relationship, while segmented linear regression does to some degree.

One can also go one step further than using segmented linear regression and apply nonlinear regression. Nonlinear regression is able to capture nonlinear relationships in different ways based on the type of the nonlinear model that is solved. Some nonlinear relationships can be transformed to a linear model by using transformation functions. For example, in some situations it may be possible to transform a nonlinear function to a linear function that can be solved by using linear regression by taking the logarithm of both sides of the nonlinear function. When a nonlinear function cannot be transformed to a linear one, the Gauss-Newton algorithm is used to determine the parameters of the nonlinear function. This procedure is similar to the one used in the previously discussed neural networks model.

2.1.4 Decision trees

Predictions can also be performed by constructing decision trees. Decision trees consist of nodes that each contain their subset of data which represent a unique combination of data features. A node and its subset can be split into one or more child nodes that contain data subsets of their parent node. The node that is split forms a decision node and contains a classification rule to be able to navigate to its child nodes. The goal is to reach the leaf node, a node that contains no children, representing the data relevant for your prediction. This leaf node is reached by following the classification rules associated with the decision nodes of a decision tree.

Decision trees can be applied as a classifier as well as a predictor for a continuous variable. For classification purposes classification trees are used, while regression trees are able to form continuous predictions. When a leaf node is acquired, the prediction for classification trees is equal to the most commonly appearing class within the data subset of the leaf node, while for regression trees it is common practice to retrieve the average value of the predictor variable within the subset.

Constructing a decision tree is done based on a training data set. The tree that is built will have a prediction accuracy of 100% on this data set. A problem that needs to be considered when building decision trees is the issue of overfitting. The generated tree is built completely towards the characteristics of the training data set, and using such a tree on another data set will likely hurt the prediction performance. To prevent the usage of a tree that is overfitted towards a training set and that is not generally applicable, two types of methods are used to reduce the size of the generated tree. One way is to stop further building a tree after a certain criteria is met. An example of a decision tree method that prematurely stops tree growth is CHAID [17]. By using a chi-square statistical test the method determines whether node splits are possible that improve the purity (the degree of dispersion of data contained in node) by a statistically significant amount.

Another way is to first build a full tree and to then remove parts of the tree by using a pruning method. The popular CART algorithm [6] prunes a fully constructed tree by removing the nodes that contribute the least to the prediction performance of a tree. The algorithm keeps track of the prediction performance on a validation set of trees of varying sizes. The tree that is associated with the best prediction performance for that validation set is selected and applied for predictions.

It might occur that a good split happens below a bad split. A method that stops tree growth might not make the first bad split, and thus would then never have the chance to make the good split. Applying a pruning method overcomes this problem, as with a pruning method the good split would be made.

In order for a decision tree to perform well, a large data set is generally required. To tackle this problem, Breiman and Cutler proposed to use a random forest model [5]. Within this model, first multiple decision trees are constructed based on a data set. When a forecast then needs to be formed, a prediction is retrieved from individual decision trees and based on these individuals predictions the forecast is produced.

One can imagine that building a decision tree for a large data set requires many computations. For a data set with many features, the data set can be divided into many subsets that represent unique features, and thus the built tree would become large. If one would then apply a pruning method as well, computation time can rise quickly. Similar to training neural networks, building regression trees thus may not be a good choice for use within the Power TAC domain due to the limited computation time restraints.

From the prediction methods that we examined we prefer methods which require no extensive training of parameters in order to perform well, since in Power TAC forecasts should be provided on short-time intervals. If a Power TAC broker agent is not able to perform its actions in time, that agent will be at a disadvantage against other brokers who do execute their actions timely. Thus, neural networks and decision tree models that require training to be able to perform proper forecasts, are not preferred. An option is to limit the amount of training in order to save computation time by using heuristics; another is to use methods that require no training to perform proper forecasts.

2.2 Performance measures

Next to the prediction method that is used, it is also important to determine how exactly to evaluate the performance of such a method. Four types of forecast-errors can be distinguished: scale-dependent, percentage-error, relative error, and scale-free error metrics [15]. We will discuss examples of each of these four metrics.

2.2.1 Scale-dependent metrics

Scale-dependent metrics encompass the most basic types of evaluation measures. These measures simply rely on measuring the difference between predictions and their target values. Metrics include the mean absolute deviation (MAD), which measures the average absolute error of forecasts and their target values, the mean square error (MSE), and the geometric mean absolute error (GMAE). Scale-dependent metrics are only useful when applied on a single series of data, because, as their name implies, they are scale dependent and cannot be compared across different series of data.

2.2.2 Percentage-error metrics

The authors of [12] discuss a number of evaluation measures that use a percentage error in some way. These measures are unit free and scale-independent and therefore can be applied across multiple data series to compare prediction performance. The metric that is used most commonly for calculating the percentage error of a prediction is the Mean Absolute Percentage Error (MAPE), which is displayed by Formula 2.1.

$$MAPE = \frac{F - A}{A} \tag{2.1}$$

where F is the forecast value, and A is the actual value. Instead of the actual value, the forecast can also be used as the denominator, but the actual value is most commonly used.

The outcome resulting from the MAPE measure is intuitive and immediately shows the user how close a prediction is to its target value. However, when using this approach, problems arise when the actual value, when used as the denominator, is zero. In our domain of energy consumption and production, it might happen that there is no energy consumption during certain timeslots. A way to avoid dividing by zero is to use an arbitrarily small number instead, but this may result in very large percentage errors, since the MAPE performance measure does not contain an upper limit. This can greatly bias the MAPE, since a few large absolute percentage errors may have a large impact on the mean due to their large size.

Due to the disadvantages of MAPE, multiple alternatives of the MAPE performance measure have been applied in the past. One alternative is the symmetric MAPE (sMAPE). Rather than dividing by the actual value, the absolute difference is divided by the average of the prediction and the actual value as displayed by Formula 2.2.

$$sMAPE = \frac{F - A}{(A + F)/2} \tag{2.2}$$

Due to the way sMAPE is computed, the outcome will never be larger than 200% and thus contains an upper limit. However, even though the measure is named 'symmetric MAPE, it is not symmetric [11]. The outcome of two predictions having the same absolute deviation from an actual value can be evaluated differently. For example, if one prediction is 1100 and the other is 900, while the actual value is 1000, then the sMAPE outcomes will be as follows:

$$\frac{1100 - 1000}{(1000 + 1100)/2} = 9.5\%$$

and

$$\frac{900 - 1000}{(1000 + 900)/2} = 10.5\%$$

The above example shows that a prediction below the actual value is handled differently than a prediction above the actual value. Another disadvantage of sMAPE is that regardless of the prediction, the outcome will always be equal to 200% when the actual value is zero. The sMAPE measure thus also is not capable of dealing with zero values properly.

Another measure that is proposed as an alternative to MAPE is the MAD-to-Mean performance measure [14]. The prediction performance of a forecast is computed by dividing the absolute prediction error by the average actual value in the data set. This measure overcomes the problem that arises with MAPE when one needs to deal with zero values, while still being almost as intuitive. However, when the data set contains a large variance among the data values, using the average actual value as the denominator is not recommended, since certain predictions could be considered better than others while in actuality they are worse. In our domain the actual customer consumption may differ greatly over different timeslots, and using the average value as the denominator may give the wrong picture about certain predictions.

A simple alternative for the MAPE is the Percent Better measure, which computes the percentage of individual predictions that are more accurate than predictions of a benchmark forecasting method. This method gives insight into whether a method is consistently more precise than another, but it does not provide information about the degree to which a method is better or worse [4]. For example, when a method is slightly more accurate than another method for many predictions, but contains far larger errors for the other predictions, this is not visible through the use of the Percent Better method.

2.2.3 Relative error metrics

The authors of [4] proposed to apply evaluation measures based on relative errors rather than measures based on percentage errors. Such measures directly compare the errors obtained from one method to the errors acquired from another. For example, the mean relative absolute error (MRAE) is defined as follows:

$$MRAE = mean(\frac{e_t}{e_t^*}) \tag{2.3}$$

where e_t is the error of a prediction model and e_t^* is the error of another benchmark prediction model; typically a simple random walk method where a prediction is equal to the most recent observation. Instead of using the mean, one can also employ the median (median relative absolute error, or MdRAE) or geometric mean (geometric absolute error, or GMRAE). The authors of [4] prefer the GMRAE when the aim is to calibrate the parameters of a prediction model since this method has a good sensitivity and in order to assess a performance change in a model, the sensitivity of a forecast measure should be good. When the goal is to select a method from a number of methods, they claim that MdRAE or MdAPE is preferred, because these methods have good outlier protection.

Next to using measures that compute individual relative errors, another option is to employ relative measures that directly compare the results of the same measure for two different methods. For example, Armstrong and Collopy [4] proposed the cumulative relative absolute error, which is defined as:

$$CumRAE = \frac{MAE_a}{MAE_b} \tag{2.4}$$

where MAE_a is the mean absolute error for prediction method a, and MAE_b is the one for prediction model b. Methods other than MAE can also be applied. Measures such as CumRAE provide a direct comparison between two methods that is simple to understand.

2.2.4 Scale-free error metrics

A third alternative is the Mean Absolute Scaled Error (MASE), introduced by Hyndman and Koehler [16]. This measure uses scaled errors that are based on the average absolute error of a naive forecast method. The authors defined MASE as follows:

$$MASE = mean(|q_t|), \text{ where} \tag{2.5}$$

$$q_t = \frac{e_t}{\frac{1}{n-1}\sum_{i=2}^{n}|Y_i - Y_{i-1}|} \tag{2.6}$$

where e_t is the error of the t-th prediction, Y_i represents an actual data value, and q_t is a scaled error. When a scaled error is smaller than 1, this means that the prediction from the evaluated method is more accurate than the average prediction from the naive method, while an error larger than 1 means it is worse. The MASE is an intuitive evaluation measure that clearly shows whether a method is better than the naive method and also to what degree this is the case.

We prefer performance measures that are intuitive and immediately show to which degree a prediction deviates from its target value. Relative error metrics and scale-free error metrics are good to see whether a method performs better than another, but they do not give us insight into the deviation of a prediction from its target. Even though our evaluation comprises a comparison between our proposed state-based method and a regression tree approach, we do not prefer to use these two metric types due to their lack of intuitiveness in terms of individual prediction accuracy. Using scale-dependent metrics is also not an option, since we will perform our evaluation using multiple time series and a scale-dependent metric does not allow for good comparison between performances on multiple series. Thus, percentage-error metrics remain. The main disadvantage of these metric types results mostly from not being able to handle a zero denominator. However, if we can ensure that the data used for our evaluation does not contain zero values, this disadvantage vanishes.

Chapter 3

Power TAC Framework

In this chapter the characteristics of the Power Trading Agent Competition (Power TAC) are described. First, in Section 3.1, we will discuss the setup of Power TAC by covering the actions a broker agent can take. Then we will go into more detail about the Power TAC pilot competition that was held at the International Joint Conference of Artificial Intelligence (IJCAI) 2011 in Section 3.2. Since the focus of this thesis is on the short-term prediction of customer consumption and production, the consumption behaviour of the customers during the pilot competition will be reviewed. Only a general outline of the Power TAC game will be provided. For more details on the Power TAC competition one can read the Power TAC game specification [19].

3.1 A general overview of Power TAC

Power TAC is a trading agent simulation in which energy broker agents compete against each other with the aim of obtaining high monetary profits. Each broker agent is able to attract customers in order to generate energy consumption and production. As other brokers compete in the competition, a broker continuously needs to act in several ways to outperform the other brokers. Figure 3.1 shows the activities which a broker is involved with in Power TAC.

The activities depicted in Figure 3.1 are interactions that occur in one timeslot. In Power TAC, a timeslot represents a one-hour period in which a broker can perform actions and in which the Power TAC server provides the broker

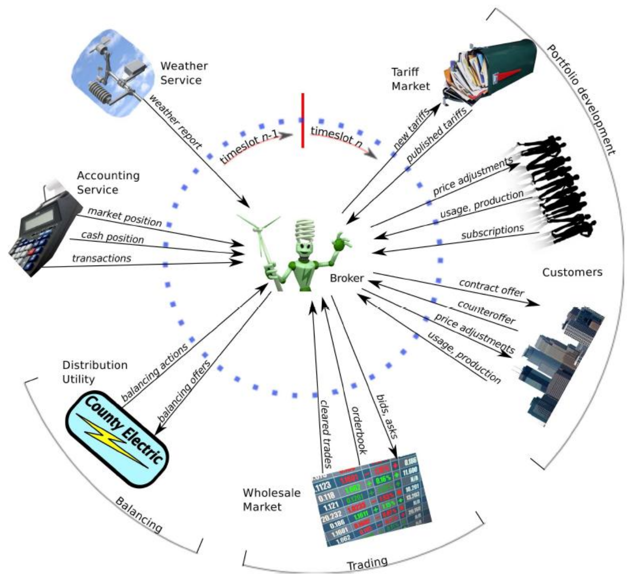

Figure 3.1: An overview of the activities for a Power TAC broker agent. Source: Power TAC game specification [19].

with information. In real-time each timeslot takes five seconds. A broker needs to perform all the desired actions for one hour within these five seconds, which means that all actions of a broker are required to have short computation times.

In the figure a clear distinction is made between multiple aspects of the game. First there is portfolio development, which mainly consists of interacting with customers. Each broker is able to design new tariffs, which are then published in the tariff market. The tariff market contains all the tariffs that have been offered by all participating broker agents. The tariffs may include varying rates depending on the hour of energy consumption. For example, a tariff may be offered which has low rates during night-time hours, but higher rates during daytime. Previously offered tariffs may also be revoked or adjusted.

For the Power TAC version that we use for our thesis, the version from July 2011, only anonymous small customers are included. The type of customer that have been used in the pilot competition are as follows:

- Households; customers with a typical residential consumption pattern.
- Offices; customers that have flat consumption during working hours, and low consumption during other hours.

- Factories; customers with similar behaviour as offices, but with higher volatility.

The customers are able to select a tariff published on the tariff market. When a customer is attracted to a certain tariff of a broker, it will consume or produce for the broker that is associated with that tariff. Customers may represent a larger population, such as a village. For example, within the pilot competition held in July 2011 customers were present that represent 1,000 or even 10,000 households. Each individual of a population can decide to select a different tariff. Thus, when a customer represents a population of 10 individuals, this population can be distributed over 10 different tariffs. Customers are able to change the tariffs they use during the competition, and therefore brokers continuously need to manage their portfolio of offered tariffs.

Figure 3.1 also illustrates the possibility of contract offering with customers. This however is not implemented in the Power TAC version we base our thesis research on. In the future, Power TAC will also contain large customers next to the small customers. With these large customers the broker agents need to negotiate contracts to attract them. The tariffs will not play a role within the interaction process between brokers and these large customers.

When a broker attracts customers, the total consumption and production of the customers also needs to be taken into account. Power TAC namely contains the Distribution Utility, which will penalize brokers when they have an imbalance between energy consumption and production. The larger the imbalance, the larger the fee will generally be. The Distribution Utility will provide penalties during each timeslot. As a result, the aim of a broker agent is to consistently get a balance between energy consumption and production. A very important component of a broker thus is to accurately predict future energy imbalance in order to properly take actions to reduce or eliminate the imbalance. In order to aid brokers with performing accurate predictions on customer consumption, Power TAC provides every broker with historical data containing 14 days of consumption data for each customer.

A way of counteracting energy imbalance on the long term is to manage your tariffs in such a way that the consumption and production of energy within your customer portfolio is at least expected to be close to balanced. Thus, when a broker has an oversupply of energy, the broker needs to offer a tariff that is likely to attract customers that consume that oversupply to become more balanced again.

When the energy usage of the customers within your portfolio is expected to be imbalanced on the short-term (up to 23 timeslots), acting on the wholesale market is one way for brokers to reduce that imbalance. On this market, the brokers are able to offer bids and asks in order to buy or sell energy. The wholesale market is open for the next 23 timeslots, which means that the broker can resolve expected imbalance for up to 23 hours ahead through trading. The wholesale market uses a periodic double auction mechanism. This means that buyers and sellers can simultaneously submit respectively bids and asks for the next 23 timeslots during the current timeslot. After all bids and asks for a certain timeslot are submitted, the clearing price of all bids and asks can be determined. Figure 3.2 displays an example of how the clearing price is determined within Power TAC. This figure is directly retrieved from the Power TAC game specification [19].

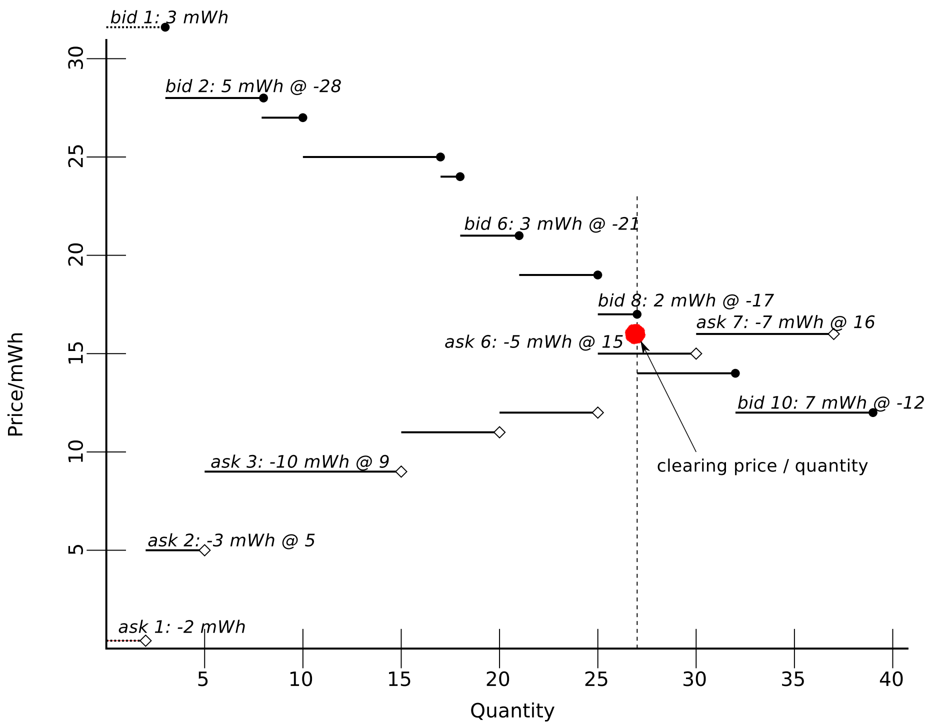

Figure 3.2: Market clearing example. Source: Power TAC specification [19].

In Figure 3.2 a number of bids and asks are illustrated. The clearing price that is associated to the bids and asks is determined by finding the intersection of these bids and asks. The bids with a price higher than the clearing price and the asks with a price below the clearing price are executed. The highest bids and lowest asks are cleared first. Either the last cleared bid or the last cleared asks is mostly executed only partially. In the figure, bid 8 and ask 6 are cleared last, ask 6 is only cleared partially. The clearing price is 16.

After the processing of portfolio development, trading on the wholesale market, and the balancing of the Distribution Utility, some information is published to the broker, such as the current cash position of the broker and weather reports. Since weather forecasts are not included yet in the Power TAC version we use as a basis for our thesis, the weather information cannot be used yet to predict the consumption and production of a broker's customers. However, in the future this aspect will play an important role, because customer behaviour will be dependent on the weather.

Another aspect that is currently not implemented yet within Power TAC is the possibility of partly controlling the consumption of customers through the use of reserving capacity. In the future brokers are able to temporarily shut down or start devices in order to consume less or more and reduce the imbalance between total energy consumption and production.

3.2 The customers of Power TAC

In July 2011 the Power TAC pilot was held at the International Joint Conference of Artificial Intelligence (IJCAI) 2011. The pilot competition contained a total of four simulations. However, only the data of one simulation is available. Based on the data of this particular game, we perform an analysis of the customer behaviour during this simulation.

A total of five broker agents participated in the game:

- EUREBA; from Erasmus University Rotterdam.
- CrocodileAgent, from the University of Zagreb.
- IAMPower; from the University of Southampton.

- Mertacor; from Aristotle University of Thessaloniki.
- defaultBroker; a simplistic default broker that is built in within the Power TAC environment.

The simulation contained a total of ten different customers. All of these customers are small customers that use the tariffs that are offered by the brokers. The customers and their characteristics can be observed in Table 3.1.

Table 3.1: The customers within the Power TAC pilot competition that was held in July 2011.

Name	**Type**	**Population**
Village 1	Consuming household	8
Village 2	Consuming household	8
DowntownHouseholds	Consuming household	10,000
MorewoodHouseholds	Consuming household	1,000
HighlandBusinesses	Consuming office	100
UniFacilities1	Consuming office	1
UniFacilities2	Producing office	1
JennywoodPark	Producing office	1
WindmillCoOp	Producing office	50
SunnysideSolar	Producing factory	2

The first aspect we analyse is the distribution of customer populations over the different brokers. This way we can get an insight into whether certain customers are volatile in the tariffs they use, or whether they stay at the same broker. The distributions of the population of each of the ten customers are depicted in Appendix A.

The *defaultBroker* agent has attracted all customers during the start of the game, but quickly loses almost all of them after one timeslot. For some of the customers the distribution of brokers over their population is straightforward in the sense that only one broker fully attracts the customer population for the remainder of the competition. For the *Village 2* customer (Figure A.2) the *CrocodileAgent* broker attracts its population of 8, while the *IAMPower* broker fully attracts three of the four producers of the game, namely *JennywoodPark*, *SunnysideSolar*, and *UniFacilities2*. The other producer, *WindmillCoOp*, is also mostly attracted by the tariffs of *IAMPower*, but also partly uses tariffs of *Mertacor*.

The attraction of the customers *Village 1* (Figure A.1) and *UniFacilities1*

(Figure A.6) is a duel between *Mertacor* and *CrocodileAgent*. The full population of both of these customers changes between tariffs of these agents, although not often. Thus, what seems to be noticeable for the behaviour of most of the customers is that they stay at a certain broker and are not very sensitive to change, at least for the examined pilot competition.

There are however customers of whom the population is more distributed among multiple brokers. For the *DowntownHouseholds* customer (Figure A.3), 50% of the population uses tariffs of the *EUREBA* broker at the beginning of the game, while 40% of the population is attracted to *CrocodileAgent* , and both *IAMPower* and *defaultBroker* attract 5%. After 8 timeslots *IAMPower* loses its 5% of attracted customer individuals to *Mertacor*. At timeslot 32, the distribution of the customer populations changes greatly. *EUREBA* loses 25% of its customers to *CrocodileAgent*, and the default broker swaps position with *Mertacor*. The largest change takes place during timeslot 37, when *Mertacor* moves from 0% attraction to 75% attraction. *EUREBA* again loses some customers, approx. 20%, while *CrocodileAgent* loses its leading position. After some more changes, the distribution becomes stable until the end of the game.

The pattern for the *MorewoodHouseholds* customer is remarkably similar, as is depicted in Figure A.4. The distribution and changes of the brokers over the customer population are almost exactly the same, only the timeslots at which tariff changes occur are different. Both customers also reach an equilibrium in distribution relatively quick, with the *DowntownHouseholds* customer reaching stability at timeslot 55 and the *MorewoodHouseholds* customer at timeslot 61. Since the competition consisted of a total of 312 timeslots, the customers were insensitive to switching brokers for the largest portion of the competition.

The only customer of which the population does not reach full stability is *HighlandBusinesses* (Figure A.5). At the start of the game, many brokers are involved with this customer, but after approx. 40 timeslots the attraction of the this customer ends up in a duel between *CrocodileAgent* and *Mertacor*. Figure A.5 clearly shows this on-going duel. Occasionally *Mertacor* loses 10% of its attracted population to *CrocodileAgent*, but wins them back again quickly.

Next to the distribution of customer populations over the different brokers in the pilot competition, we also investigated the total consumption or production of the different customers. The figures associated to the energy usage of the customers can be observed in Appendix B. Vertical lines are included in the figures at a rate of 24 timeslots to be able to better compare the consumption between different days.

The customers *Village 1* and *Village 2* are similar in behaviour and both have a clear pattern in their consumption, as is illustrated by respectively Figure B.1 and Figure B.2. The two customers frequently have two large peaks in their daily consumption. The peaks also happen to occur during the same hours. Due to their relatively stable consumption pattern, it is easier to predict when large peaks in consumption will arise.

The *DowntownHouseholds* customer shows very odd consumption behaviour. Figure B.3 clearly shows that the consumption during the first 48 timeslots differs greatly from the consumption in the later timeslots. The reason for this large change in total consumption is unknown, but it is likely to be caused by an error within the Power TAC software environment. The consumption of the *DowntownHouseholds* customer after timeslot 48 seems to be consistent, with a larger total consumption during the end and beginning of each day.

The consumption or production of the other customers that are represented in Appendix B seems to be more randomized. There are no clear patterns and the consumption or production has a high volatility. For the *WindmillCoOp* and *SunnysideSolar* customer (Figure B.7 and Figure B.8) this behaviour could be explained by weather influences. However, since weather forecasts are not yet included in Power TAC and thus cannot be used for consumption prediction, the consumption of these two customers in particular becomes unpredictable.

The consumption of two of the customers, *JennywoodPark* and *UniFacilities2*, are not included in Appendix B. The reason for this is that the production of both of these customers always was zero during the complete pilot competition.

Based on the customer consumption depicted in Appendix B, we find that the consumption of the varying customers does not seem to represent their customer types as was stated in the Power TAC game specification. No different consumption pattern seem to be apparent between the consumption of varying customers. For example, for *MorewoodHouseholds* and *HighlandBusinesses*, one would not able to notice that the first is a household consumer while the other is an office based on the consumption depicted in Figure B.4 and Figure B.5. It can be expected that in the future, as Power TAC develops further, the consumption will be more representative of the customer types.

Through a discussion of the components of Power TAC and the customer consumption of Power TAC customers that were present during the Power TAC pilot competition, we have gained insight into what to expect in terms of customer consumption prediction. We described and illustrated that customers contain a population of which each individual can use a different tariff. From the Power TAC game specification we can derive that the consumption will be influenced by the customer types and power types of customers, as well as the time of consumption. All of these aspects help in defining the prediction model we will describe in Chapter 4,

Chapter 4

State-based Prediction Framework

In this chapter a short-term prediction method that can be applied within the framework of a Power TAC broker agent is described. This prediction method is used for the purpose of forecasting the short-term imbalance of energy consumption and production for a broker agent. In Section 4.1 the general components of a broker agent are first described to show the role of such a short-term prediction method within a global framework of a Power TAC broker agent. The method will mostly be evaluated by utilising self-generated data. The reasons for using this self-generated data and the setup to build the data is covered in Section 4.2. Then, in Section 4.3, the short-term prediction method is described in detail.

4.1 The components of a broker agent

In Chapter 3 we already described the interactions of a broker agent within Power TAC. We will shortly repeat the main actions of such an agent and focus on the role of forecasting short-term energy imbalance to gain insight into the importance of these predictions within a broker's framework. Figure 4.1 displays a general overview of a broker's main tasks and the interdependencies between them.

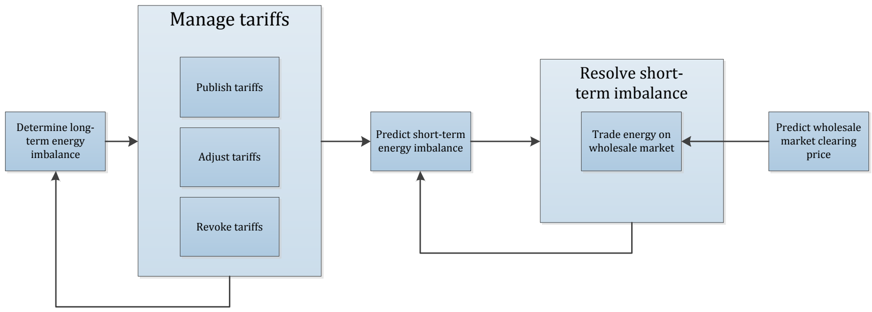

Figure 4.1: An overview of the general tasks of a Power TAC broker agent.

The first task for a broker is to publish tariffs to attract customers. The attracted customers will consume or produce energy. The total consumed and produced energy of a broker's customers needs to be balanced, or otherwise the broker will receive a penalty. Therefore, the imbalance between consumption and production continuously needs to be taken into account. Dealing with energy imbalance can be done in two different ways. First there is the long-term imbalance that needs to considered, and secondly there is the short-term imbalance.

The long-term energy imbalance can be taken hold of by properly managing your tariffs. If a broker observes that its customers are consistently consuming a lot more energy than they produce, a logical action would be to either attract more producers or to remove consumers from your portfolio. By publishing or revoking the right tariffs, the long-term energy balance can be controlled. The better a broker manages its portfolio, the lower the long-term imbalance will be. However, as balanced as a broker's portfolio may be in expectation, the consumption and production will mostly not be completely balanced on the short-term. Thus, a broker also needs to determine what imbalance in energy to expect on the short-term to solve the short-term energy imbalances as well.

Accurately predicting the short-term imbalance between energy consumption and production within the portfolio of a broker agent is an important part of a broker's framework. It can be perceived as the motor for all actions that can be taken on the short-term to prevent imbalance fees. Without it a broker will suffer more losses. Based on the predicted short-term imbalance, different actions can be set in motion. If a shortage of energy is predicted, then one would try to acquire energy, while a predicted surplus of energy would mean the opposite. In the version of Power TAC from July 2011 the only action that may be performed by a broker to resolve predicted imbalances on the short-term is to trade energy on the wholesale market. To trade on the wholesale market, a broker requires a desired amount of traded energy based on the predicted imbalance, next to the wholesale market clearing price to know at which prices to trade. However, the focus of our thesis is not on the actions that resolve energy imbalance, but on forecasting the short-term imbalance in a proper way so that such actions can be set in motion.

4.2 Generating customer data

In Chapter 3 we discussed the energy consumption of customers within the Power TAC pilot competition that was held at the International Joint Conference of Artificial Intelligence (IJCAI) 2011. We noticed that the consumption of these customers does not seem to be satisfying. For most customers, the consumption had no pattern. Also, the consumption did not represent the customer types of the customers. For example, there was no clear difference between customer offices and customer households. Due to the nature of the data of the Power TAC pilot competition, it seems futile to apply prediction models to determine future imbalances of energy consumption and production using this data. As the main goal of this thesis is to design a prediction method and we would like to test this data on useful data, we decided to generate our own consumption data that does represent the type of a customer and that does have a pattern.

For the ten customers that were present in the pilot competition, the consumption of two of them still seems to be appropriate to use. *Village 1* and *Village 2* have a recurring consumption pattern that seems to represent their customer type. Both of these customers are household consumers with a total population of 8 households. The Power TAC pilot competition contained more household customers, as well as office and factory customers. We will generate consumption data representing the same three customer types. Building data that represents these three kinds of customer types of the pilot competition will result in a group of heterogeneous customers, which is interesting to evaluate.

The behaviour of the three types of customers (household, office, and factory) is determined based on both their descriptions within the Power TAC game specification [19], and the data provided by Madison Gas and Electric (MGE) [1], an energy company that operates in the area of Madison, Wisconsin. MGE provides information on the general consumption behaviour of several groups of customers, such as hotels, offices, and multifamily residences.

In the game specification of Power TAC the behaviour of households in terms of energy usage is described as 'typical residential consumption'. In Figure 4.2 the typical behaviour of households according to MGE is depicted.

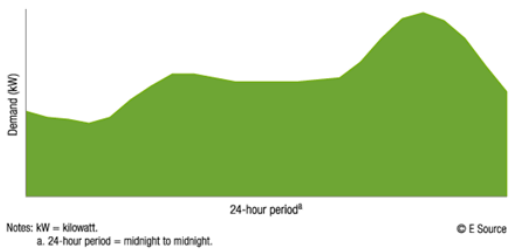

Figure 4.2: Typical consumption of a multifamily residence according to Madison Gas and Electric. Source: MGE [1].

Figure 4.2 illustrates the total energy consumption of a household over the course of a day. From the figure we learn the following characteristics:

- The consumption is low during the night.
- There is increasing consumption during the early morning until 7 to 8 AM.
- There is a relatively stable consumption during late morning and early afternoon.
- In the late afternoon the consumption starts to increase significantly. Eventually a peak is reached at about 8 PM.
- After 9 PM the consumption starts to decrease greatly until about 12 PM to 1 AM.

The main features of residential consumption are the two peaks in consumption. A small peak arises in the morning, and a much larger peak during the evening. These consumption patterns seem plausible. In the morning people awake and start to use more energy, while in the late morning and early afternoon people generally are away for work or school and thus less energy is used. When people return later on the day, more energy is used again. This behaviour is clearly represented in Figure 4.2.

The second customer type we will generate data of is the office. According to the Power TAC game specification, 'offices have a typical flat consumption during working hours', and 'limited consumption at other times'. The figure obtained from MGE, Figure 4.3, shows us a similar pattern.

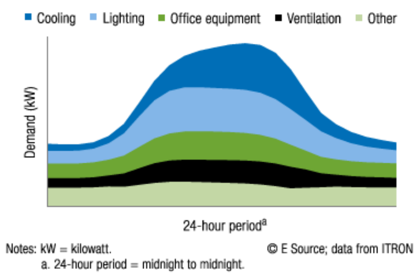

Figure 4.3: Typical consumption of an office according to Madison Gas and Electric. Source: MGE [1].

From early evening to early morning there is a relatively small amount of consumption. Starting in the early morning, the consumption quickly increases until about 9 AM, after which the consumption is relatively stable, with a slight increase, until about 3 PM. Then the consumption starts to decrease significantly until it reaches a more stable consumption at about 8 to 9 PM.

A third type used within the Power TAC pilot competition are factories. Factory consumption is described in the game specification as being 'similar to office consumption but with greater magnitudes and more variance'. For this reason, to generate data representing factory consumption or production, we use a pattern that has a high similarity to the one depicted in Figure 4.3. The difference is that the increase and decrease in energy usage during respectively early morning and late afternoon will be larger to represent a larger magnitude.

We will now describe in detail how we build the consumption data for the customers. The creation of this data is mainly characterized by the use of basic patterns to assure that the generated consumption data represents the different type of customers. For the three types of customers we initially design two base patterns: one for weekdays and one for weekends. These patterns are based on the characteristics described previously. The base patterns consist of 24 predefined proportions that each represent an hour. Mapping these proportions to an image would result in a pattern similar to the ones illustrated in Figure 4.2 or Figure 4.3, depending on the customer type. Using the base patterns, we start to generate simple consumption data for different customers. For each of the customers, we define one base value. This base value determines the overall magnitude of a customer's consumption.

For each customer, a maximum random factor is also defined. The higher this factor, the more generated consumption is likely to deviate from the base patterns. The trend of a customers consumption is taken into account as well. If the consumption for a certain timeslot is higher than usual, then the consumption for the timeslot that succeeds that timeslot has a larger probability of also being higher than usual.

Each customer within Power TAC represents a larger population. For example, the customer *Village 1* from the Power TAC pilot competition that was described in Chapter 3 represents a total of eight individual households. The energy consumption of a customer c for timeslot $z \in h, h = \{0, 1, 2, ..., 23\}$, is generated as follows:

$$v_z^c = p_z^c \cdot b^c, \text{ where} \tag{4.1}$$

$$p_z^c = g_{z,w}^c \cdot (1 + \tau_z^c), \tag{4.2}$$

$$\tau_z^c = \alpha \cdot random(d_c) + (1 - \alpha) \cdot \tau_{z-1}^c, \text{ and} \tag{4.3}$$

$$\tau_{-1}^c = 0 \tag{4.4}$$

where d_c is a random factor in the interval $[0, 100]$ for customer c, $random(d_c)$ is a randomly computed proportion within the interval $[-\frac{d_c}{100}, \frac{d_c}{100}]$, $g_{z,w}^c$ is the base proportion associated with customer c related to timeslot z that falls within week part w, τ_z^c represents the relative increase of the to-be computed proportion for timeslot z over the base proportion $g^c_{h,w}$, p^c_z is the computed proportion for customer c at timeslot z, b^c is the base value associated to customer c, and v^c_z is the generated consumption in kWh of customer c for timeslot z. There is also a weight value α within the interval $[0, 1]$. The higher this weight, the more impact the randomly generated proportion $random(d_c)$ has on the generated proportion τ^c_z, and the lower the influence of the proportion from previous timeslot τ^c_{z-1}. The value for α we used to generate data is 0.3. This value is selected so that the consumption trend has a higher impact on the created consumption values than the random factor that is included.

What we are still missing compared to the consumption data from a Power TAC simulation are the tariffs. So far we have simply generated the total consumption of a customer per timeslot. In an actual Power TAC game, however, the consumption of the full population of a customer can be distributed over multiple tariffs. We thus need to find a way to properly divide the consumption values we have generated over a number of tariffs and associate a consumption value with each tariff. Also, we would like to have different consumption behaviour based on which tariff is used by a customer. To decide how we can manage varying consumption depending on tariffs, we must first know how tariffs are defined in Power TAC.

Each tariff is composed of 24 rates that each are associated to a specific hour. For the Power TAC version from July 2011, these rates are fixed, and are not subject to variable pricing. If a customer is willing to change its consumption behaviour depending on the used tariff, the smartest option for that customer would be to consume more energy on hours for which the tariff rate is low, and to consume less on hours where the price is higher. Based on this thought, we alter the consumption generated with Formula 4.1 by redistributing the consumption dependent on the used tariff rates of customer individuals. The total consumption for timeslot z and tariff f for customer c is measured as follows:

$$q^c_{z,f} = \sum_{i \in c \cap f} (1-\rho) \cdot \frac{v^c_z}{|c|} + w^i_{r_z} \cdot (\rho \cdot \frac{v^c_{d_z}}{|c|}), \text{ where} \tag{4.5}$$

$$v^c_d = \sum_{z \in d} v^c_z, \text{ and} \tag{4.6}$$

$$w_{r^i_z} = \frac{(\sum_{j \in d_z} r^i_j) - r^i_z}{\sum_{k \in d_z}((\sum_{j \in d_z} r^i_j) - r^i_k)} \tag{4.7}$$

where i is the collection of individuals that are part of the population of customer c and that use tariff f during timeslot z, ρ is the proportion of the daily consumption that is redistributed over the 24 hours of a day, d_z represent day d to which timeslot z is associated, $w_{r^i_z}$ is the weight of the rate r^i_z that is used by individual i during timeslot z. This weight is determined by comparing it to the other rates that are used by individual i during day d. The lower rate r^i_z is in comparison with the others, the more consumption will be shifted towards timeslot z. If $\rho = 0$, then no consumption is shifted towards other hours.

Depicted in Figure 4.4, 4.5, and 4.6 are the generated consumption values for three different types of customers. Figure 4.4 represents the consumption of a household consumer, in Figure 4.5 the consumption of an office consumer is illustrated, while Figure 4.6 shows the production of factory producer. The random factors we used to generate these three consumption sets are respectively 8, 15, and 75.

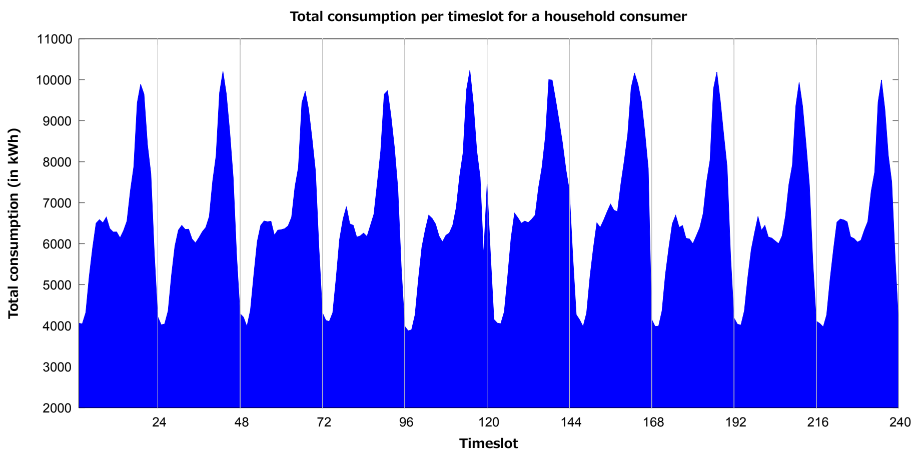

Figure 4.4: The generated consumption for 10 days for a household customer.

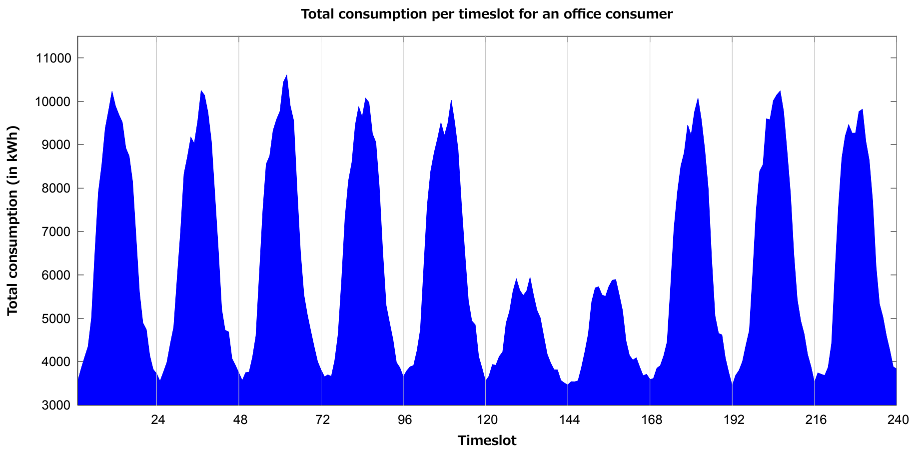

Figure 4.5: The generated consumption for 10 days for an office customer.

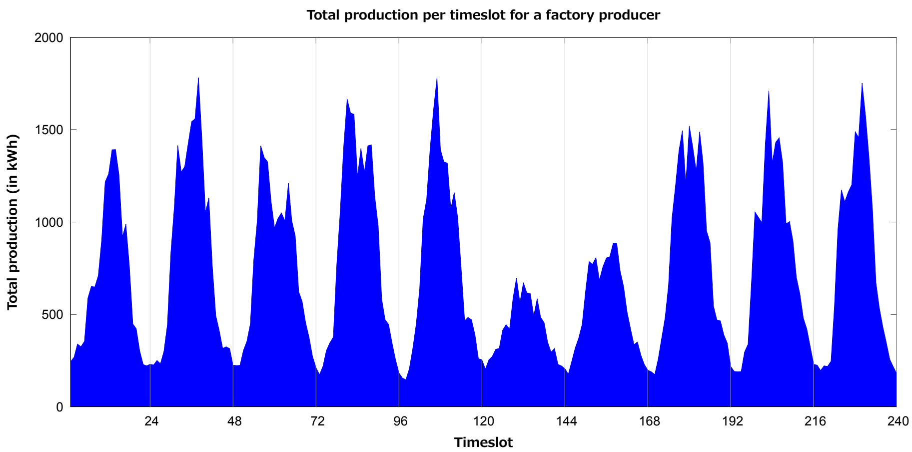

Figure 4.6: The generated production for 10 days for a factory customer.

Figure 4.4 illustrates that the generated consumption for a household consumer has a pattern similar to the consumption depicted in Figure 4.2. Figure 4.6 clearly shows that the production of the factory customer is less smooth than the consumption of the office customer due to the higher random factor. The gap between the lower and higher production of the factory customer is also larger than the one of the office customer, which distinguishes the factory customer further from the office customer.

4.3 Building the state-based prediction model

One of the tasks of a broker agent within Power TAC is to predict the imbalance of energy consumption and energy production within the customer portfolio. Short-term predictions need to be performed to forecast the energy imbalance of the upcoming timeslots. Then, based on these predictions, actions can be taken to resolve the forecasted energy imbalance. In Section 4.3.1 we describe the general setup of our prediction method based on a number of guidelines and assumptions. Then, in the following sections we outline in detail the different steps of our prediction method.

4.3.1 The guidelines of a prediction

The short-term energy imbalance within Power TAC is dependent on the behaviour of the participating customers. Thus, to design a short-term prediction algorithm we first need to find out how the behaviour of customers is influenced on the short-term. Based on the Power TAC game specification [19] we find that the following aspects / features may have an impact on the energy consumption or production of customers.

- **Customer** All customers have a specific load profile that determines how they use energy. The load profile can be divided into two parts: the customer type, and the power type. A customer always is linked to one customer type, but may be associated with multiple power types.

  The customer type characterizes the general energy usage behaviour of a customer. Customer types include households, offices, and factories in the version of Power TAC from July 2011. All of these types have consumption behaviour that distinguishes them from each other. Offices have typical flat consumption throughout working hours, and lower consumption outside working hours, while households represent typical residential consumption behaviour. Thus, the customer type of a customer affects his behaviour, and needs to be taken into account for more accurate predictions.

  There are a number of power types in Power TAC that represent the power flow of a customer. For example, a customer may support the 'consumption power type, which means that power flows from the grid to the customer. Another power type is 'production, for which the energy flows from the customer to the grid. This production power type is further split into sub types that allow differentiation of power sources. These subtypes include solar production and wind production. However, the current version of Power TAC only uses a limited number of different power types, as weather is not integrated into the environment yet. Naturally, the power type of a customer has a great impact on the prediction of short-term energy imbalance, since it determines whether a customer is consuming or producing energy.

Each customer has a customer type, at least one power type, but also a population. One single customer may represent a village where a number of people are living, or a large office that has hundreds of employees. Even if the customer type and the power types are shared between different customers, the consumption can still differ between the customers.

- **Part of the week (week / weekend)** Customers have other behaviour depending on whether they are consuming on a weekday or during a weekend. Offices will likely have lower energy consumption during the weekend, while households may experience an increase in consumption.

- **Time** Like the part of the week, the time on the day also affects energy consumption of customers. During the night, offices are closed and people within households are sleeping, and thus will not consume much energy. Furthermore, the tariffs that are offered by brokers may contain different prices depending on the time of consumption, which is likely to influence customer behaviour.

- **Tariff** Another aspect that has an impact on consumption and production of a customer is the tariff that is used. For example, tariffs may include time-of-use rates or weekday/weekend rates to encourage certain behaviour, such as higher consumption, during specific time periods.

- **Weather** The behaviour of customers is also influenced by the weather. Future Power TAC simulations may contain producers that use wind based production, or solar based production. Furthermore, the consumption of customers is likely to be influenced by the temperature. Customers might use more power for heating devices during periods in which the temperature is low, i.e.

We need to take into account the aforementioned aspects to create an accurate prediction algorithm. For this reason, we decided to design an approach that specifically considers the varying aspects. Our approach is based on using so-called states that each represent a unique combination of data features. The states are defined based on three different features. They are as follows:

- Customer
- Part of the week
- Tariff

The number of states that need to be used is dependent on the amount of values that the above three features can take. The number of customers and tariffs differ per Power TAC competition. The customers are known at the start of the game, and do not change during the competition. The amount of used broker tariffs vary over the course of a competition.

Weather is not included as a feature that is used to define states. The data model presented in Section 4.2 is based on the Power TAC version from July 2011, which did not include weather features for customer consumption, and as a result this feature is not included in our prediction model as well. With the three listed features, we define different states that each represent their own pool of data. Let us show how to retrieve a state with a brief example.

Example 1 - Retrieving a state *We currently are in timeslot z_t on a regular weekday. One of our customers, customer c, is an office which supports the 'consumption' power type. This office uses our tariff f. We now need to forecast its consumption for the next timeslot, which is z_{t+1}, which also falls on a weekday. Thus, the state s for the to-be-predicted timeslot represents the following features:*

- *Customer: c*
- *Part of the week: week*

- *Tariff: f*

State s thus contains all past consumption from customer c that took place during weekdays when using tariff f.

Next to the listed aspects there are three important assumptions we can make and that have an influence on how we build our prediction model.

- The consumption of a specific timeslot is partly dependent on the consumption of the most recent past timeslots.

- At the start of a competition, the broker receives for each of the customers two weeks of consumption data.

- Customer consumption roughly follows a daily pattern.

The magnitude of the predicted consumption value for a future timeslot is likely to be dependent on the consumption of the most recent timeslots, i.e. the higher the consumption during the most recently occurred timeslots, the higher the consumption for a future timeslot is likely to be. Therefore, using a prediction model that relies on the use of time series is a good approach.

The second assumption allows us to initiate the to-be-built prediction model with data at the start of a competition. Since we have this historical data already available at the beginning of a Power TAC competition, it is possible to create a model dependent on historical data that is capable to start well. The general setup of our prediction model will therefore be modelled around the use of states to divide data into different data pools, and then use for each prediction a specific state in combination with a historical data approach.

As described and explained in Section 4.2, the consumption of each customer roughly follows a daily pattern. The consumption is based on the hour of consumption. On certain hours the consumption will consistently be larger than others, and vice versa. This means that it could be useful to take into account the pattern of each customer in the setup of our prediction model by considering the consumption associated to the hour of the future timeslot we require a prediction for.

Now that we have defined states and described some assumptions that influence our prediction model, we illustrate the general sequence of forming predictions using these states. Figure 4.7 depicts the steps that are taken to form a prediction during one timeslot.

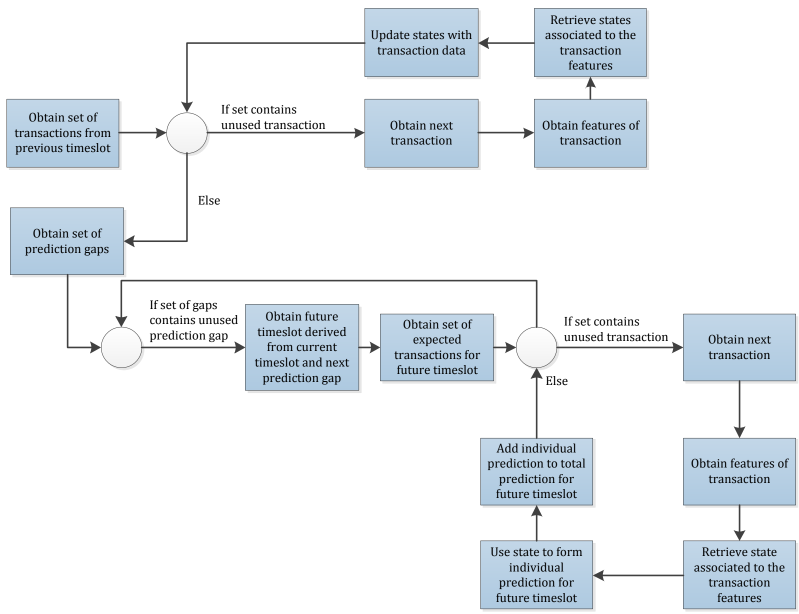

Figure 4.7: The sequence of steps that are used to form a prediction during a single timeslot.

As can be observed in Figure 4.7, there are quite some steps that need to be taken to form short-term imbalance predictions. First, storage of the newly received past data takes place by updating states that are related to the data. Then, we form predictions for future timeslots based on expected transactions for those timeslots and by employing the states that are associated to those expected transactions. We now describe in detail seven different steps that are part of the storage and prediction process of our state-based prediction method.

4.3.2 Step 1 - Storing past data

The very first step is to store the newest data available, so that the states in our prediction model are updated with recent information. During each timeslot of a Power TAC game, a broker may receive a list of tariff transactions. One tariff transaction specifies the consumption or production in kWh during a specific timeslot of a specified customer using a specified tariff. When performing the prediction process during a timeslot, the first step is to obtain the tariff transactions from the previous timeslot. The consumption and production values associated to these transactions can then be stored in the states that are related to them.

Assume that the following transactions related to previous timeslot are received, and that the previous timeslot is related to a weekday [1]:

- **Transaction** t_1
  - Customer: c_1
  - Tariff: f_1
  - Customer count: 1000
  - Consumption (in kWh): 1,100
- **Transaction** t_2
  - Customer: c_2
  - Tariff: f_1
  - Customer count: 20
  - Consumption (in kWh): 1,500
- **Transaction** t_3
  - Customer: c_1
  - Tariff: f_2
  - Customer count: 100
  - Consumption (in kWh): 80

The data contained in these transactions will be stored in several states. Based on the features of each tariff transaction we know exactly which states need to be updated by adding the new tariff transaction data. If we consider transaction t_1, we update the states that represents tariff f_1, customer c_1 and

[1] Note that the data is simply used for explaining purposes and does not directly represent actual data.

weekdays. If a state represents another customer, another part of the week, or another tariff, then that state is not updated.

The transaction quantities of tariff transactions are always published as the total consumption of a specified number of individuals. Since the number of individuals associated to a specific tariff transaction can differ between transactions, we store the transaction quantities per individual and not in total, so that the stored quantities are universal and can be compared to each other. For the three listed transactions the following values would be stored:

- Transaction t_1: $\frac{1{,}100}{1{,}000} = 1.1$
- Transaction t_2: $\frac{1{,}500}{20} = 75$
- Transaction t_3: $\frac{80}{100} = 0.8$

The way in which our data is stored is influenced by the setup of our prediction process. To execute a prediction, an aspect to take into account is the past energy usage for a specific hour. Figure B.1 shows that customer *Village 1* always consumed significantly more during the period from 8:00 to 8:59 during the pilot competition. If one would not consider the unusually high consumption that is associated to this particular hour, but only the consumption of the most recent timeslots, the predictions associated to customer *Village 1* for timeslots related to this hour will most likely always be too low. To perform predictions, we thus would like to directly use the consumption data associated to a specific hour.

When forming a prediction for a future timeslot that falls within a certain hour, a reasonable approach is to first examine how high or low the energy usage within the most recent past timeslots was, and then consider this information in conjunction with information about the consumption for the future timeslot's hour. If the consumption in past timeslots is low and the consumption for a specific hour is always high, then we do not want our prediction to be small as well due to the information obtained from the past timeslots. Rather, we would like to base our prediction for that hour to be based on the relative magnitude of the consumption during the most recent timeslots. If the consumption was relatively high during past timeslots, our prediction would be relatively higher, and vice versa.

Since we will take into account the hour of consumption when forming a prediction, we also consider this aspect when storing data within states. Assume that the set of k states is $S = \{s_1, s_2, ..., s_k\}$. For each $s_i \in S$ a set is included that consists of 24 lists of data values $\{L_0^i, L_1^i, L_2^i, ..., L_{23}^i\}$. Each of these lists contains the consumption values associated to one specific hour, and thus $L_h^i = \cup_{j=1}^{j=d} n_{j,h}^i$, where d is the current day's index, $n_{j,h}^i$ is the consumption value at hour h, day j for state s_i. Thus, consumption values are stored for each hour of the day. When a tariff transaction is received, we know for which hour the transaction quantity should be stored. If we then receive a transaction for a certain hour, we store the transaction data in a list of values that are all associated to that hour.

Because we would like to include the relative magnitude of a consumption value compared to other consumption values that fall within the same hour, all 24 lists contained within a state are ordered from low to high. Each time a new data value is stored for a certain hour, we maintain this order. By keeping track of the order of each of the 24 groups, we can easily obtain information about whether consumption for a specific timeslot was higher or lower than usual for the hour that timeslot falls within. Figure 4.8 illustrates how the individual transaction quantity from transaction t_1 is stored within a state and how we can retrieve information about the relative magnitude of that quantity.

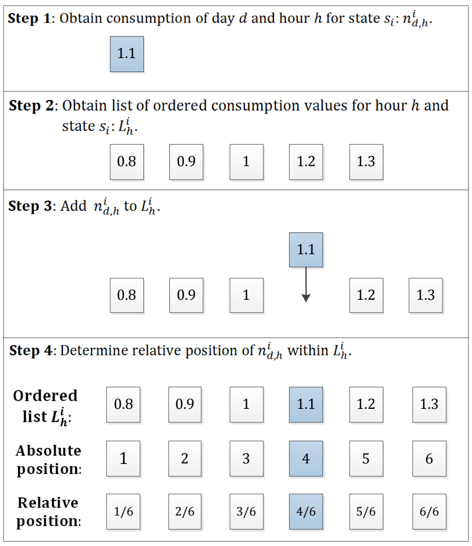

Figure 4.8: Storage process of a consumption value within state s_i.

Figure 4.8 depicts the storage process of the consumption value of transaction t_1 within the state s_i. The transaction quantity per individual of transaction t_1 was 1.1 kWh and is the quantity that we store in the state. The second step is to retrieve from s_i a list of values belonging to the hour of consumption of transaction t_1, which is hour h. The list L_h^i related to hour h and s_i contains all consumption values of past timeslots within the state that are associated to hour h. For our predictions we will also take into account the relative magnitude of the consumption. In order to easily retrieve this magnitude, the lists of consumption values associated to an hour are all ordered in ascending order. Figure 4.8 shows that L_h^i contains the following values: $\{0.8, 0.9, 1, 1.2, 1.3\}$. In comparison with the values within this list, the consumption of 1.1 kWh is the third highest value, and thus should be placed fourth in the list. Because L_h^i contains a total of six values, the relative position of the individual transaction quantity of transaction t_1 is $\frac{4}{6} \approx 0.67$. These relative positions will play a large role in forming our short-term imbalance predictions. Algorithm 1 describes the process of storing transaction data from a specific timeslot to update states.

Algorithm 1 Updating states

```
Input: timeslot z_c, hour related to timeslot z_c: h_{z_c}, set of transactions associated to timeslot z_{c-1}:
  T_{z_{c-1}}
Output: The storage of the most recent historical data
  obtain empty set of to-be-updated states for timeslot z_c: S_u
  for each transaction t ∈ T_{z_{c-1}} do
      obtain consumption of t per individual: n
      obtain set of features of t: F_t
      obtain set of states that represent F_t or solely subsets of F_t: S_{F_t}
      for each state s_{F_t} ∈ S_{F_t} do
          compute average consumption for state s_{F_t} for timeslot z_c
          if s_{F_t} ∉ S_u then
              add s_{F_t} to S_u
          end if
          perform Algorithm 4 with respectively n, z_c, and h_{z_c} as input
      end for
  end for
  for each state s_u ∈ S_u do
      obtain average consumption for state s_u for timeslot z_c: n̄_{z_c}
      obtain ordered list of values for hour h_{z_c} and state s_u: L^u_{h_{z_c}}
      add n̄_{z_c} to L^u_{h_{z_c}}
      obtain relative position of n̄ within L^u_{h_{z_c}}: p
      obtain list of relative positions per timeslot: P
      add p to P
  end for
```

4.3.3 Step 2 - Determine future timeslot

During the prediction process within one timeslot we perform short-term imbalance forecasts for multiple future timeslots. One possible action to resolve predicted energy imbalances is to trade energy on the wholesale market. On this wholesale market, one can buy and sell energy for up to 23 timeslots ahead. For this reason, we perform predictions for up to 23 timeslots into the future. We refer to the number of timeslots between the current timeslot and the timeslot for which a prediction is performed as the 'prediction gap'. Figure 4.9 displays for which timeslots predictions are performed when the current timeslot is z_c and the set of prediction gaps is $\{g_1, g_2, ..., g_n\}$.

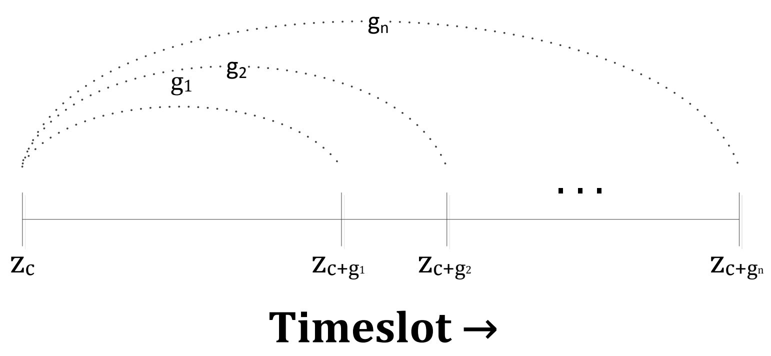

Figure 4.9: The timeslots for which predictions are performed given that the current timeslot is z_c and the set of prediction gaps is $\{g_1, g_2, ..., g_n\}$.

Depending on the number of gaps we define, the number of timeslots for which forecasts are done becomes larger and as a result the number of predictions grows as well. For example, if the set of prediction gaps G is $\{1, 2, 3, 6, 12, 23\}$ and the current timeslot is z_1, then we would perform predictions for timeslots z_2, z_3, z_4, z_5, z_{13}, and z_{24}. The prediction gap we use throughout the remainder of this chapter is simply referred to as g, while the future timeslot is notated as z_{c+g}, which takes place g timeslots after current timeslot z_c.

4.3.4 Step 3 - Determine expected transactions

To form the energy imbalance forecast for the future timeslot z_{c+g}, we must know which customers are expected to consume and produce within the broker's portfolio and also which tariffs they will use. Assume that the following customers are expected to consume for our broker in timeslot z_{c+g}.

- 1000 individuals related to customer c_1 using tariff f_1
- 20 individuals related to customer c_2 using tariff f_1
- 100 individuals related to customer c_1 using tariff f_2

To form a prediction for timeslot z_{c+g}, we compute individual predictions for each of the above three expected transactions. The sum of these predictions is the forecast for timeslot z_{c+g}. In future references, we refer to the above three expected transactions as e_1, e_2, and e_3. To obtain the individual predictions associated to each expected transaction, we must first know how to retrieve states that are representative for a transaction and can then be used to form a forecast.

4.3.5 Step 4 - Determine relevant past timeslots

Before we retrieve states to start performing our predictions, we first decide which past timeslots are relevant for our forecast. One parameter setting in our prediction model is the number of past timeslots that are taken into account to form a prediction. We dub this parameter as m.

Figure 4.10 depicts a timeline containing current timeslot z_c and other timeslots which form the relevant timeslots for our prediction.

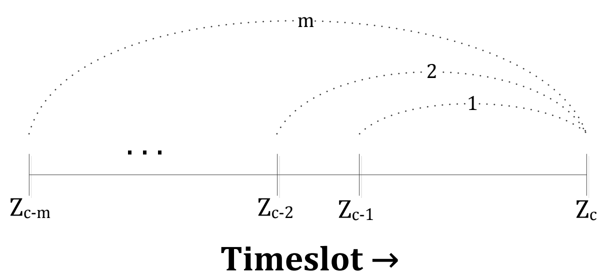

Figure 4.10: A timeline representing the relevant past timeslots for a prediction that occurs at timeslot z_c when the last m timeslots are considered relevant.

4.3.6 Step 5 - Retrieve state for prediction

For each of the three expected tariff transactions in Section 4.3.4 we need to acquire one state. However, selecting a state based on a transaction is not simply a process of picking the state that represents the features of that transaction. The reason for this is that the state that represents those features may represent a too specific combination of transaction features and as a result may not contain the proper amount of data to form a prediction. For example, when a state s_i is required for a prediction for hour h, but no consumption has taken place for that hour in the past for state s_i, then we are unable to form a prediction using the data contained in state s_i. To counter this issue, a good option is to look for other similar states that do contain the data required for prediction. But first we define when a state is deemed as having too few data. The only reason is the following one:

- The hour of the timeslot for which the prediction is performed contains less than η values.

The value of η is one that is free to choose by the user. The higher this number, the more reliable predictions from a specific state will be, but at the same time the chance that other states have to be used for predictions grows as well. When other states are used for prediction, the forecast may also be negatively influenced. The question is where to cross the line between only using states that contain a decent amount of data and using other states that contain enough data, but are not completely representative for the features of the expected transactions.

When a state does not have enough data, a replacement state needs to be used for prediction instead. We now describe in detail the steps and components that are related to the process of finding this replacement state. The most important aspect in finding a similar state is to discover which features of an expected transaction are relevant for our prediction and should be considered in the process of finding a new state. In total we take three features into account that distinguish states: the tariff, the part of the week, and the customer. However, it is possible that a customer does not have different consumption between weekdays and weekends. In that case it would be disadvantageous to split the consumption data based on the part of the week and use states that represent both that customer and a part of the week. If dividing data based on that feature would not be done, we would be able to use relevant data from a bigger pool of data. Therefore, at the start of a Power TAC simulation, we determine for each customer whether we should use states that represent a part of the week.

At the start of a Power TAC competition, we receive historical data for each customer that represents the consumption for the last 14 days prior to the start of the competition. Based on this historical data of a customer, we can determine if the part of the week plays a significant role in the consumption behaviour of that customer. Whether the part of the week has a significant influence on the energy usage is determined by performing a matched pair statistical t test for each customer. This test is used frequently in medical research, i.e. to examine the effects of medicines [3]. In a similar way, we can investigate the impact of the part of the week to the consumption of customers.

First, we retrieve two states that both represent different customer data: one state that contains all past consumption during weekdays, and one that contains all weekend consumption. In the matched pair t test, we compare 24 pairs in which each pair consists of the average consumption on a specific hour during weekdays, and the average consumption on the same hour during weekends. The values for these pairs are retrieved from the two states. When according to the test the values for a customer differ significantly at a 95% confidence level, then we still use the part of the week as one of the data features for that customer. If there is no significant difference, we do not split the data based on the part of the week. The algorithm for determining whether there is a significant difference is displayed in Algorithm 2. Note that this algorithm can be used to determine significant difference between any pair of states.

From the three features that we use, splitting up data based on the tariff is the most likely reason why a state could have a lack of data. Since at the start of a simulation we receive 14-day historical data for each customer including consumption for both weekdays (10 days) and weekends (4 days), we will have enough data based on the other two features unless the η parameter is set above four (in this case, states representing weekends would not contain enough

Algorithm 2 Determining significant difference between states

Input: a state s_a, another state s_b
Output: whether s_a and s_b are significantly different at a 95% conf. level
  obtain lists from s_a, each associated to a different hour: $L_0^a, L_1^a, ..., L_{23}^a$
  obtain lists from s_b, each associated to a different hour: $L_0^b, L_1^b, ..., L_{23}^b$
  $totaldif = 0$
  $sumSqrdDif = 0$
  for $h = 0 \rightarrow 23$ **do**
    obtain average consumption associated to L_h^a: $\bar{L_h^a}$
    obtain average consumption associated to L_h^b: $\bar{L_h^b}$
    $dif \leftarrow \bar{L_h^a}$ - $\bar{L_h^b}$
    $totaldif \leftarrow totaldif + dif$
    $sumSqrdDif \leftarrow sumSqrdDif + dif^2$
  end for
  $sampledif \leftarrow \frac{totaldif}{24}$
  $totalDifSqrd \leftarrow totaldif^2$
  $standarderror \leftarrow \sqrt{\frac{sumSqrdDif - \frac{totalDifSqrd}{24}}{23}} / \sqrt{24}$
  $t \leftarrow \frac{sampledif}{standarderror}$
  if $t <$ -2.06866 or $t >$ 2.06866 **then**
    s_a and s_b are significantly different
  else
    s_a and s_b are not significantly different
  end if

data, since only four consumption quantities per hour would be stored). To find a state that is similar to a state that does not contain data required for prediction, we therefore first examine if there is another state that represents the same customer and part of the week, but a different yet similar tariff.

The tariffs in Power TAC are composed of rates. These rates are the main distinction factor between tariffs. Each rate within a tariff defines the price per kWh for a certain number of hours. For example, there may be a rate with a price of 0.2 that is used from hour 23 to hour 6 on each day. Each tariff is composed of rates in such a way that all hours have a rate associated with them. For the Power TAC version from July 2011, these rates are fixed, and are not subject to variable pricing. To measure the similarity between two tariffs, we simply compare the hour rates between the tariffs.

Consider tariff f_1 and tariff f_2 that respectively have the following rates associated to them: $\{r_0^{f_1}, r_1^{f_1}, ..., r_{23}^{f_1}\}$ and $\{r_0^{f_2}, r_1^{f_2}, ..., r_{23}^{f_2}\}$, where r_h^f is the rate of tariff f associated to hour h. Assume that we require a prediction for timeslot z_{c+g} and that we take into account the consumption of the m most recently occurred timeslots. The required state associated to tariff f_1 does not contain the required data to form our prediction. How to find out if tariff f_2 is similar enough to tariff f_1 to use it for our forecast instead of tariff f_1? We simply directly compare the rates of both tariffs that are relevant for our prediction.

For our forecast we use the past m timeslots to form a prediction and we are in current timeslot z_c. This means that the collection of past timeslots we consider for our forecast is $\cup_{i=1}^{i=m} z_{c-i}$. To compute the dissimilarity between tariffs we now use the rates that are associated to the timeslots within this collection and the future timeslot z_{c+g} for which a prediction is made. The degree to which a timeslot z has an impact on our forecast is defined as w_z. The higher the weight associated to a timeslot, the more important that timeslot is for our forecast and the more important the tariff rate associated to that same timeslot, and thus we use the same weights in measuring the dissimilarity between two tariffs as we do for forming a forecast. When past tariff rates are similar, but the tariff rate for the future timeslot of prediction is not, then it is still not preferable to use the other tariff for prediction, as the rate associated with the future timeslot is simply not representative enough. Therefore, we also compute the dissimilarity related to the future timeslot and assign a relatively large weight to this future timeslot. We apply a weight of 0.5 for the weighted dissimilarity of past timeslot rates, and a weight of 0.5 for the future timeslot's rate. Formula 4.8 displays how the dissimilarity is measured:

$$\delta_{f_1,f_2}^{Z,z_{c+g}} = 0.5 \cdot dif_{f_1,f_2}^{z_{c+g}} + 0.5 \cdot \sum_{z_p \in Z} w_{z_p} \cdot dif_{f_1,f_2}^{z_p}, \text{ where} \tag{4.8}$$

$$dif_{f_1,f_2}^{z} = \frac{|r_{h_z}^{f_1} - r_{h_z}^{f_2}|}{|r_{h_z}^{f_1}|} \tag{4.9}$$

$$\sum_{z \in Z} w_z = 1 \tag{4.10}$$

where Z is the set of past timeslots relevant for the prediction, z_{c+g} is the future timeslot for which the prediction is made, w_z is the weight of the rate for timeslot z, and r_h^f is the rate of tariff f for hour h. The obtained dissimilarity is a percentual value.

When the dissimilarity between two tariffs is acquired, we can determine whether one tariff is similar enough to be used instead of the other to retrieve a state. If $\delta_{f_1,f_2}^{Z,z_{c+g}}$ is below a specific threshold, tariff f_2 may be used as a replacement for tariff f_1. When the retrieved dissimilarity is larger, then f_2 is deemed to different, and it is not used. The same customer might have used more than just one other tariff. In this case dissimilarity values need to be computed for multiple tariffs, and the selected tariff is the one with the smallest dissimilarity that is also below the specified parameter value. If for this tariff the associated state also does not contain the data required for prediction, then the tariff with the second smallest dissimilarity lower than the threshold is picked, etc. If there are no states that have a dissimilarity value below the specified threshold and that contain enough data, then no tariff is used.

If no similar states with enough data can be retrieved, a good option is to retrieve data from a more global state that represents less amount of features. The more global a state is and the less aspects it takes into account, the more likely that state is to contain data required for doing forecasts. For example, there may be a state s_x that contains all the consumption data of customer c, for which the part of the week is 'week' and the tariff is f. For a more global state s_y we could omit one of the three characteristics, such as the tariff. This state s_y would then contain consumption during weekdays of customer c for all tariffs that it has used, and not only data associated to tariff f, which means that the amount of data for this state s_y is at least as large as the amount of data for the more specific state s_x.

To use more global states when a specific state does not have the proper data available, we need to define parent-child relationships between states to be able to navigate from one state to another. For this reason we use a tree that contains the used states and their parent states. Since there are only three relevant features that are taken into account, and since the part of the week may not be included for certain customers, the state tree remains small.

Figure 4.11 shows the general structure of the state tree to demonstrate how one can use data of more global states.

At the bottom of the tree presented in Figure 4.11 are the most specific states, states that take into account the largest amount of data features. Whenever we require a prediction, we start with one of these bottom-layer states. If one observes that there is not enough data available to form a prediction and there is no similar state to use, one moves a layer upwards to a more global state. This process is repeated until a state contains the data required for forecasting, which in the worst case is a state that only represents a customer.

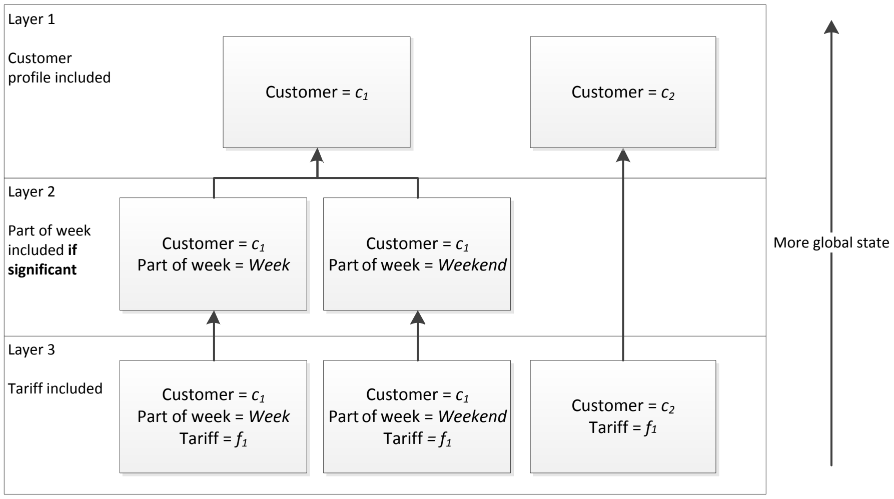

Figure 4.11: Moving from specific states to more global states.

In Figure 4.11 the states for two customers are displayed. For one of these two customers the part of the week does not have a significant impact on the consumption or production. As a consequence, the parent state of the state from layer 3 is a state from layer 1, and layer 2 is skipped. All states, except the states from layer 1, have one parent to maintain simplicity and allow easy navigation. Algorithm 3 shows in detail how a state associated to a specific tariff transaction is retrieved.

Assume that from the three expected transactions listed in Section 4.3.4 tariff f_1 has been frequently used before and states associated to this tariff contain enough data required for prediction, while tariff f_2 is used for the first time. Tariff f_2 has a dissimilarity with f_1 that is below the dissimilarity threshold and is the least dissimilar tariff of f_1. Also presume that customer c_1 has significantly varying consumption between weekdays and weekends, while customer c_2 does not. Which states would we retrieve under these circumstances for the three transactions?

- **For e_1:** the state that represents customer c_1, part of the week 'week', and tariff f_1. This state represent the same features as e_1 and contains enough data, so it can be used for prediction.

Algorithm 3 Retrieving a state for prediction

Input: timeslot z_c, transaction t, prediction gap g, min. no. of required observations η
Output: The state s_{used} associated to transaction t

```
obtain tariff f_t, customer cust_t, and part of week wp_t from t
s_week ← state that represents f_t, cust_t, and wp_week
s_weekend ← state that represents f_t, cust_t, and wp_weekend
if s_week and s_weekend are significantly different according to Algorithm 2 then
    s_1 ← state that represents f_t, cust_t, and wp_t
else
    s_1 ← state that represents f_t and cust_t
end if
z_{c+g} ← timeslot to predict for
h_{z_{c+g}} ← hour related to timeslot z_{c+g}
if |L^1_{h_{z_{c+g}}}| ∈ s_1 ≥ η then
    s_used ← s_1
else
    f_sim ← tariff that is most similar to f_t
    s_o ← other state that represents f_sim and same other features as s_1
    if |L^o_{h_{z_{c+g}}}| ∈ s_o ≥ η then
        s_used ← s_o
    else
        s_p ← parent state of s_1
        repeat
            if |L^o_{h_{z_{c+g}}}| ∈ s_p ≥ η then
                s_used ← s_p
            else
                s_p ← parent state of s_p
            end if
        until |L^o_{h_{z_{c+g}}}| ∈ s_o ≥ η
    end if
end if
```

- **For e_2:** the state that represents customer c_2, and tariff f_1. In our example, customer c_2 does not have varying behaviour between weekdays and weekends, and thus the part of the week is not taken into account.

- **For e_3:** the state that represents customer c_1, part of the week 'week', and tariff f_1. Since tariff f_2 is used for the first time, states that represent this tariff do not contain enough data required for prediction. In our example tariff f_1 has a dissimilarity with f_2 that is lower than the dissimilarity threshold, and is used instead.

4.3.7 Step 6 - Forming an individual prediction

After step 5, the question is how to compute a prediction when we have acquired a state. We now describe in detail the process of forming an individual prediction. Or in other words, the procedure to make a prediction associated to one expected transaction for only one future timeslot. The expected transaction we perform a prediction for is transaction e_1. For this particular prediction we employ the state s_{e_1} we acquired by following the steps from Section 4.3.6, and use the prediction gap g.

The first step we take when forming an individual prediction within one state is to determine the relative magnitude of the consumption we try to predict. For convenience reasons, we define this relative magnitude as the 'timeslot trend'. This timeslot trend is based on the relative magnitude of the consumption of the most recent past timeslots. To determine the timeslot trend, we will compute a value between 0 and 1, where the magnitude is larger as the outcome is closer to 1. The higher the timeslot trend, the higher the prediction will be, and vice versa.

The values from the past timeslots are used to form a prediction for timeslot z_{c+g}. When the consumption for the previous timeslot is higher, one would expect the consumption for the next timeslot also to be larger. However, the variable g can take multiple values, ranging from 1 to 23. When a prediction is formed for a timeslot that occurs 23 hours ahead, the relationship between the most recent past consumption and the future consumption is likely to not be so clear. Since the predictions are not only performed for the next timeslot, we are required to design a prediction method that is capable of forecasting energy imbalance further than one timeslot into the future. Thus, to setup a prediction we would like to consider the relationship of past data to future data.

We require the relationships between the set of past timeslots $\{z_{c-1},\ z_{c-2},\ ...,\ z_{c-m}\}$ and future timeslot z_{c+g}. The relationship between timeslots is determined by the hours to which they belong, as well as the gap between the current timeslot and the timeslot for which a prediction is made. In each relationship the input values are the values associated to the earlier timeslot, and the output values are linked to the later timeslots. We define h_{z_t} as the hour which timeslot z_t belongs to. When the input value belongs to timeslot z_x and the output value to timeslot z_y and the prediction is performed g timeslots in advance, we use the data set that contains all input values and associated output values of h_{z_x} and h_{z_y} that had a gap of g timeslots in between them. For convenience, I will define such a mapping as $\phi : g, h_{z_x} \rightarrow h_{z_y}$.

The following functions are required to compute the timeslot trend and depend on the user-selected m-value:

- $\phi : g, h_{z_{c-1}} \rightarrow h_{z_{c+g}}$
- $\phi : g, h_{z_{c-2}} \rightarrow h_{z_{c+g}}$

- ...
- $\phi : g, h_{z_{c-m}} \rightarrow h_{z_{c+g}}$

Now the question is how to determine the relationships between the input and output values. We use a simple linear regression model, which computes a linear function based on a list of input and output values. The function between a list containing the input values $(x_1, x_2, ..., x_n)$ and a list containing the linked output values $(y_1, y_2, ..., y_n)$, is as follows:

$$f(x) = \beta \cdot x + \alpha, \text{ where} \tag{4.11}$$

$$\beta = \frac{\sum_{i=1}^{n}((x_i - \mu_x) \cdot (y_i - \mu_y))}{\sum_{i=1}^{n}(x_i - \mu_x)^2}, \text{ and} \tag{4.12}$$

$$\alpha = \mu_y - \beta \cdot \mu_x \tag{4.13}$$

where μ_x is the average of the input values, and μ_y is the average of the output values. The function that is acquired by Formula 4.11 minimizes the sum of residuals between the data points and the line associated to the obtained linear function. Each time we receive new transaction data we update linear functions by recomputing α and β. Algorithm 4 describes in detail this process.

Algorithm 4 Updating linear functions

Input: timeslot z_t, the hour h_{z_t} related to z_t
Output: updates linear functions by adjusting their α and β values

```
obtain set of prediction gaps: G
obtain number of past timeslots taken into account for prediction: m
obtain relative magnitude related to timeslot z_t: p_{z_t}
for each prediction gap g ∈ G do
    for i = 1 → m do
        z_past ← z_{t-i-g}
        if consumed during timeslot z_past then
            obtain relative magnitude related to timeslot z_past: p_{z_past}
            obtain hour related to timeslot z_past: h_{z_past}
            obtain list of relative magnitudes associated to h_{z_past}: X
            obtain list of relative magnitudes associated to h_{z_t}: Y
            add p_{z_past} to X
            add p_{z_t} to Y
            xxbar = 0
            xybar = 0
            for j = 1 → size of X do
                obtain j-th value of X: x
                obtain j-th value of Y: y
                xxbar ← xxbar + (x − X̄)^2
                xybar ← xybar + (x − X̄)(y − Ȳ)
            end for
            β(g, h_{z_past}, h_{z_t}) ← xybar / xxbar
            α(g, h_{z_past}, h_{z_t}) ← ybar − β × xbar
        end if
    end for
end for
```

Although the simple linear regression method is not able to capture complex non-linear relationships, it is still a good method to apply in order to determine the relationships between consumption during different hours. Let us explain why by describing the possible relationships between the consumption of two hours. The relationship between two hours h_1 and h_2 can roughly have the following characteristics:

- If consumption for h_1 is relatively small/large, then the consumption for h_2 is also relatively small/large.
- If consumption for h_1 is relatively small/large, then the consumption for h_2 is relatively large/small.
- If consumption for h_1 is relatively small/large, then the consumption for h_2 is not of a certain magnitude and can vary in size. In this case there is no clear relationship.

For the first two relationships, the relationship is linear and can be captured by a simple linear regression model. For the third relationship, there is no clear relationship. In this case, it is hard to predict the consumption in a future hour based on the consumption in a past hour. Then a smart thing to do would be to simply take the average or median consumption associated to the future hour, since based on past data no accurate conclusions can be drawn regarding the relatively magnitude of the consumption during the future hour. Within the linear regression model, for data sets with no or a low correlation the resulting linear function will typically represent a flat or close to flat line of which the output values are close to the mean output value regardless of the input value. Thus, the simple linear regression model is a viable method to use to construct relationship functions of the consumption between varying hours. Moreover, due to the low complexity of linear regression models, the method is less computation-heavy than methods that compute non-linear relationships. This is especially important within the framework of a Power TAC broker agent, because the actions performed by a broker are required to be performed in less than five seconds per timeslot [19].

By using the simple linear regression approach, we can transform past data and use the transformed values to obtain the timeslot trend. Figure 4.12 illustrates an example of a process for obtaining a timeslot trend.

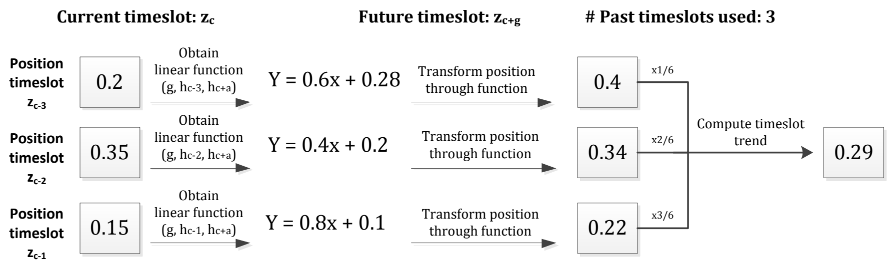

Figure 4.12: The process of computing the timeslot trend given a number of input values and linear functions.

In Figure 4.12 a total of three timeslots are taken into account for determining the timeslot trend. These timeslots are defined as timeslot z_{c-1}, z_{c-2}, and z_{c-3}. For each of these timeslots, the relative position of the consumption associated to that timeslot is depicted. Each relative position is obtained in the same way as displayed in Figure 4.8. For example, for timeslot z_{c-1} the consumption for that timeslot is compared to past consumption on hour $h_{z_{c-1}}$ to acquire the relative position of the consumption for timeslot z_{c-1}. The positions are as follows:

- Relative position consumption of timeslot z_{c-3} in distribution $h_{z_{c-3}}$: 0.2
- Relative position consumption of timeslot z_{c-2} in distribution $h_{z_{c-2}}$: 0.35
- Relative position consumption of timeslot z_{c-1} in distribution $h_{z_{c-1}}$: 0.15

The prediction in Figure 4.12 is performed for timeslot z_{c+g}, which occurs g timeslots after current timeslot z_c. The next step is to transform each of the three relative positions associated to timeslots z_{c-1} to z_{c-3} through the use of a linear function. The three relevant linear functions are depicted in the figure. We now use each relative position as an input for their associated linear function to obtain a transformed relative position. The resulting transformed relative positions are:

- $0.6 \cdot 0.2 + 0.28 = 0.4$

- $0.4 \cdot 0.35 + 0.2 = 0.34$
- $0.8 \cdot 0.15 + 0.1 = 0.22$

The transformed relative positions already give us an indication of the expected magnitude for the consumption in future timeslot z_{c+g}. The only step that is left for determining the timeslot trend is to calculate a single relative position from the three obtained transformed relative positions. To achieve this a weighted averaging approach is applied. This approach gives the transformed position associated to the most recent past timeslot the highest impact on the timeslot trend. The formula for determining the timeslot trend is:

$$timeslot\ trend = \sum_{i=1}^{m} \frac{i}{0.5(m^2+m)} \cdot p_i \tag{4.14}$$

where m is the number of past relative positions taken into account, and p_i is past relative position i. The past positions are ordered from least recent to most recent. Applying Formula 4.14 to the example from Figure 4.12 results in:

$$\frac{1}{6} \cdot 0.4 + \frac{2}{6} \cdot 0.34 + \frac{3}{6} \cdot 0.22 = 0.29$$

The acquired timeslot trend of 0.29 tells us that the expected consumption for timeslot z_{c+g} is relatively low. With the obtained timeslot trend, we can form a prediction for future timeslot z_{c+g}, which belongs to hour $h_{z_{c+g}}$. We retrieve the consumption values that are associated to this hour. Based on this list of values and the computed timeslot trend we determine our prediction. Before we show what the resulting prediction will be, we describe how we obtain a value from a list using a certain timeslot trend.

Assume that the total number of observations for hour $h_{z_{c+g}}$ is defined as n. If we then multiply the predicted relative position with n, we obtain an expected position. If this value is an integer, we can use it directly as an index to retrieve a value from the list. For example, when the obtained expected position is exactly 40, we select the 40th value in the list we examine. However, the obtained expected position is mostly not an integer, but a real value between two integers. When this is the case, we cannot directly retrieve a value from a list. Instead, we base our prediction on two values: the one associated to the highest index number not larger than (the floor value) and the one associated to the lowest index number not smaller than the expected position (the ceiling value). We compute a prediction based on the distance between the expected position and the floor and ceiling value. The closer the expected position is to the floor value, the more the obtained prediction is influenced by the floor value, and vice versa. The formula for acquiring a prediction given an expected position is as follows:

$$y(r) = (r - floor(r)) \cdot v_{ceil(r)} + (ceil(r) - r) \cdot v_{floor(r)} \tag{4.15}$$

where r is a real number, v_i is the consumption value associated to integer i, $floor(r)$ is the largest integer not higher than r, and $ceil(r)$ is the smallest integer not lower than r.

Assume the hour associated to timeslot z_{c+g} contains the following values: $\{0.7,\ 0.8,\ 0.85,\ 0.9,\ 0.95\}$. The timeslot trend that we computed is 0.29. That means that the expected position is:

$$expected\ position\ for\ timeslot\ z_{c+g} = 0.29 \cdot 5 = 1.45$$

Based on the acquired expected position for timeslot z_{c+g}, we can form our prediction. We first obtain the values associated to index 1 (the largest integer not greater than 1.45) and index 2 (the smallest integer not less than 1.45). The values associated to these indices from the list of hour $h_{z_{c+g}}$ are 0.7 and 0.8. The next step is to apply Formula 4.15 to the retrieved values to obtain a prediction.

$$prediction\ timeslot\ z_{c+g} = (1.45 - 1) \cdot 0.8 + (2 - 1.45) \cdot 0.7 = 0.745$$

However, since we stored all consumption values per individual customer, we still need to adjust this number for the customer count that is associated to expected transaction e_1. The customer count for this transaction was 1,000. Thus, we multiply the obtained consumption with this customer count. This will result in the following prediction related to transaction e_1 for future timeslot

z_{c+g}:

$$prediction\ timeslot\ z_{c+g} = 0.745 \cdot 1000 = 745\ kWh.$$

The prediction for timeslot z_{c+g} for transaction e_1 thus is equal to 745 kWh. In Algorithm 5 the same process of retrieving a prediction from a state is described.

Algorithm 5 Prediction retrieval from a state

Input: the state s_i that is used for prediction, the future timeslot z_{c+g} for which the prediction is performed, the hour $h_{z_{c+g}}$ related to z_{c+g}, the gap g between z_{c+g} and the current timeslot in number of timeslots
Output: predicted consumption related to timeslot z_{c+g} in kWh

```
prop = 0
obtain number of past timeslots taken into account for prediction: m
for i = 1 → m do
    z_past ← z_{c−i}
    obtain hour related to timeslot z_past: h_{z_past}
    obtain proportion related to timeslot z_past: p_{z_past}
    obtain linear function related to g, h_{z_past}, and h_{z_{c+g}}: f(x)
    obtain transformed proportion using p_{z_past} on f(x): y
    obtain weight of proportion y: w_y
    prop ← prop + (w_y × y)
end for
obtain list of values associated to h_{z_{c+g}}: L^i_{h_{z_{c+g}}}
retrieve consumption value in kWh from L^i_{h_{z_{c+g}}} using prop
```

4.3.8 Step 7 - Forming timeslot predictions

The prediction we obtained in step 6 is only associated with a single expected transaction. The process is repeated for every transaction in the set of expected transactions for a future timeslot, and for each value in the set of prediction gaps. In the case of the three expected transactions presented in Section 4.3.4 and a set of prediction gaps containing six values, the total number of required individual predictions would be 18, since three predictions are formed for six future timeslots. Assume that we obtained the following individual predictions for timeslot z_{c+g}.

- Prediction for timeslot z_{c+g} associated to transaction e_1: 745 kWh
- Prediction for timeslot z_{c+g} associated to transaction e_2: 1,200 kWh
- Prediction for timeslot z_{c+g} associated to transaction e_3: 68 kWh

Our prediction for the expected energy imbalance for timeslot z_{c+g} would simply be the sum of acquired individual predictions, which in this case is:

$$predicted\ energy\ imbalance\ for\ timeslot\ z_{c+g} = 745 + 1,200 + 68 = 2,013$$

Thus, the expected energy imbalance for timeslot z_{c+g} is equal to 2,013 kWh. In Algorithm 6 the general process of forming predictions is summarized.

Algorithm 6 Forming predictions

Input: timeslot z_c, hour h_{z_c} related to timeslot z_c, set of prediction gaps G
Output: Energy consumption predictions in kWh for a set of timeslots, which depends on z_c and G

```
for each g ∈ G do
    z_{c+g} ← timeslot to predict for
    T ← set of expected transactions for z_{c+g}
    for each transaction t ∈ T do
        state s_t ← Algorithm 3 with z_{c+g}, t, and g as input
        prediction y ← Algorithm 5 with s_t, z_{c+g}, h_{z_{c+g}}, and g as input
        Y_{z_{c+g}} ← total predicted consumption for timeslot z_{c+g}
        Y_{z_{c+g}} ← Y_{z_{c+g}} + y
    end for
end for
```

The aim of the broker agent now is to perform such actions that the expected imbalance for a future timeslot is resolved. Actions that can be taken by the agent to achieve this goal is to manage tariffs in such a way that the imbalance becomes smaller, for example by attracting producers. A second possibility is to buy or sell expected excess or surplus energy on the wholesale market. However, for this thesis we focus purely on the construction of a method that is capable of performing the required forecasts and not on the actions that a broker agent can take to resolve forecasted imbalances.

The acquired prediction that represents the expected energy imbalance in kWh for a timeslot in the future may provide you with information on what imbalance to anticipate, but it does not tell you anything about the certainty of the prediction.

An addition to the short-term energy consumption prediction model is to compute a prediction interval of expected consumption. Instead of providing the user solely with one number that represents the expected future consumption, we can also compute a lower limit and an upper limit. The lower and upper limit form a prediction interval. If this interval is large, the prediction is logically less certain than if the interval would be small.

When retrieving individual predictions, we compute a timeslot trend and then extract a consumption value from a list of past values that is associated to the hour of the timeslot the prediction is performed for. We can use these same hour lists to compute a lower limit and an upper limit per individual prediction as well. Summing up the lower limit of all individual predictions for a specific timeslot will then yield us the lower limit of the total prediction for that timeslot. The same mechanism holds for measuring the upper limit of the total prediction.

Measuring the lower and upper limit of an individual prediction can be performed in a rather straightforward way. Assume that X is the list of values that is related to the hour of the timeslot for which we perform a prediction, and that y is the predicted individual consumption. From X we can compute the mean and the standard deviation. Using simple t-test statistics, we can then measure a lower limit and upper limit for this single prediction. Depending on the confidence level of the prediction interval, we obtain a standard score z. The lower limit and upper limit can then be computed as follows:

$$\text{lower limit } lower(y, \sigma_h^s, z) = y - \sigma_h^s \cdot z$$

$$\text{upper limit } upper(y, \sigma_h^s, z) = y + \sigma_h^s \cdot z$$

where y is the predicted consumption, σ_h^s is the standard deviation to the consumption of hour h within state s, and z is the applied standard score.

All in all the short-term prediction method we use contains quite a number of steps. During one timeslot there are two main actions that are performed. First, the most recent transaction data is stored and states are updated. Then, for a set of future timeslots expected transactions are used to form predictions. Each individual prediction is formed by taking into account the relative magnitude of the consumption during the most recent past hours, and through the use of a simple linear regression model and a weighted method, these relative magnitudes are utilised to determine the expected relative magnitude of future consumption and with it the expected consumption.

Chapter 5

Evaluation

In this chapter the state-based prediction method presented in 4.3 is evaluated. First, in Section 5.1, a regression tree method is described. This method is evaluated for the purpose of comparison with the state-based prediction method. In Section 5.2, we define a number of evaluation scenarios that allow us to evaluate the two prediction methods under varying circumstances. Next, using the defined scenarios we analyse the state-based prediction method under different parameter setups to obtain the optimal setup for this method in Section 5.3. Last, in Section 5.4, we evaluate and compare the prediction performance of the regression tree method with the state-based prediction method using the same evaluation scenarios. Furthermore, we will examine the time performance of both prediction methods.

5.1 Constructing a regression tree

To evaluate its performance the state-based prediction method discussed in Section 4.3 will be compared to the CART regression tree approach [6]. The regression tree method shares similarities with the state-based prediction method. For both methods data is divided into smaller subsets by performing splits based on data characteristics. For this reason, a regression tree prediction approach is an interesting method to compare the state-based prediction method with.

A regression tree is a tree structure that can be used for prediction. The tree is composed of multiple nodes. Each node is associated to a set of data and may contain a prediction value. The top node, the root node, is associated with all available data. This data can be split based on a number of characteristics of the used data set. For example, a data set containing the characteristics of different cars can be divided into a subset with only cars that are red, and a subset with cars that have a different colour. The root node will then have two children nodes, one containing the red car data, and the other containing the remaining car data. These subsets can then be divided further into subsets by splitting them based on another characteristic (e.g. car brand). Eventually, once all splits are performed, a tree with leaf nodes, nodes that do not have child nodes, will be created. These leaf nodes contain prediction values that one can use for forecasting. When the outcome variable is categorical, the prediction is equal to the most occurring class in the data set of the leaf node. When the outcome variable is numerical, the prediction value is determined by taking the average value of the outcome variable for all data points in the data set of the leaf node.

The question now is how to build a regression tree from the data set discussed in Section 4.2. To construct a regression tree, we use data characteristics similar to the ones that are used for the state-based prediction method. Two features are added in comparison to the ones used for the state-based method: customer type and power type. Due to the more static setup of the state-based method, these two features were redundant in the state-based model, but for the regression tree approach they do provide value, since the order of splits is dynamic in the regression tree model. The data characteristics used in the regression tree approach are as follows:

- Customer
- Customer type
- Hour
- Part of the week
- Power type
- Tariff

Most of the used data characteristics are categorical variables. The outcome variable is the predicted energy consumption, which is numerical. The to-be-constructed tree will be a binary regression tree. This means that each node, except for the leaf nodes, will have two children, and that data sets will always be split into two subsets. Since most of the decision variables are categorical, there are many possible splits. For example, assume that there are four customers: customer c_a, c_b, c_c, and c_d. The possible splits when using customers as the split variable would then be: $\{c_a\}$ and $\{c_b, c_c, c_d\}$; $\{c_b\}$ and $\{c_a, c_c, c_d\}$; $\{c_c\}$ and $\{c_a, c_b, c_d\}$; $\{c_d\}$ and $\{c_a, c_b, c_c\}$; $\{c_a, c_b\}$ and $\{c_c, c_d\}$; $\{c_a, c_c\}$ and $\{c_b, c_d\}$; $\{c_a, c_d\}$ and $\{c_b, c_c\}$. Since customers are not the only decision variable, there are many splits to be attempted.

The only numerical variable is the hour. Although this variable can also be treated as a categorical one, we have decided to use it as a numerical variable to save computation time. There are 24 different hours and the number of possible subsets with 24 values becomes considerably large. Determining possible splits for hours is done based on threshold values. Transactions related to an hour smaller than or equal to the threshold value are stored in one subset, while the remaining transactions are stored in the other subset.

An important part of regression tree construction is to decide how to split data into two subsets. Since the predicted energy consumption is a numerical value, we can evaluate all possible splits by examining the outcome variable values in the two created data subsets resulting from the split. A term commonly used within tree construction is 'node impurity'. The node impurity represents the dispersion of the values associated to a node. If the variance between the values is larger, the impurity will also be bigger. The nodes that we use contain numerical values, and therefore we can simply take the sum of squared deviations to measure the impurity of a node. Thus, the impurity is computed as follows:

$$imp_n = \sum_{i=1}^{m}((x_i - a_n)^2) \tag{5.1}$$

where a_n is the average of all m values associated to node n, and x_i is the i-th value within the list of m values.

Now that we know how to measure the impurity of one node, it is only a small step to determine the impurity of a split. To compute the impurity of a split, we measure the impurity of the two child nodes that result from the split, and then take a weighted average of the two obtained impurity values. Equation 5.2 shows how the impurity of a split is computed:

$$imp_{split(s)} = \sum_{n \in N} \frac{imp_n \cdot m_n}{m_s} \tag{5.2}$$

where s is the node that is split, N is the set of nodes that result from the splitting of s, imp_x is the impurity of node x, and m_x is the number of values associated to node x.

We can elaborate on Formula 5.2 with a simple example. When a split results in one child node containing 10 data points and an impurity of 50, and another child node containing 15 data points and an impurity of 80, the impurity of the split is as follows:

$$\frac{50 \cdot 10 + 85 \cdot 15}{10 + 15} = 68$$

When building a tree, there may be a large number of possible splits. For each of these possible splits we compute the impurity. The split associated to the lowest impurity is then selected, so that the data is split into subsets with the lowest dispersion across data observations. Splits are attempted until the leaf nodes cannot be divided further into new leaf nodes containing smaller data sets.

Usually, when a tree is constructed to its maximum size, the tree is sensitive for data overfitting. This means that the tree works very well for the data set it is created with (a fully built tree has a prediction accuracy of 100% on the training data), but not so well for new data. For this reason, a good decision is to ‘prune’ the tree to smaller sizes. By doing this, the tree is able to perform better on new data as it does not take into account all the details that might be specific for the data set the tree is constructed with.

How do we exactly approach building and pruning the regression tree? We keep track of all tariff transactions of customers that have happened in the past. All these transactions are stored in a data set. Each timeslot we may receive new customer energy transactions and add this new data to the data set. This full data set is then divided into a training set and a validation set. For the training set we randomly select a total of approximately 60% of the transactions in the full set. The remaining 40% of the data comprises the validation set. With the training data we then construct a full regression tree in the way described before. The first regression tree will be constructed at the game start by using 60% of the 14-day historical consumption data that is provided to brokers by the Power TAC server at the beginning of a game.

The next step is to start transforming one direct parent node of leaf nodes into a leave, i.e. remove two leaf nodes so that their shared parent becomes a leaf node. One decision node which is a parent of leaf nodes thus needs to become a leaf node itself. Consider a tree T with x decision nodes. Thus, a sub tree of T is a tree with $x - 1$ decision nodes. Assume that the set of all possible sub trees of T with $x-1$ decision nodes is called S. Each tree from S has a worse prediction performance on the training set than T. From S, you therefore want to select that tree that has smallest reduction of prediction accuracy compared to T. The leaf nodes that are removed from T are the ones of which the removal results in the sub tree linked to the smallest prediction error increase for the training set. Basically, we remove the leaf nodes that add the least amount of prediction accuracy to the regression tree.

We prune to smaller sub trees until eventually the tree only consists of the root node. All the pruned sub trees are then evaluated using the validation set. The regression tree that is selected is the one that has the lowest prediction error on the validation set. With this selected tree, we then predict the transactions for the upcoming 23 timeslots. Since building and pruning regression trees in each timeslot is computationally intensive, new regression trees are not constructed every timeslot. Instead, we only build a new regression tree under the following two circumstances:

- No new tree has been built during the previous six timeslots.
- A new tariff is used in a past transaction or, when perfect information is used regarding tariff usage, is expected to be used in a future transaction.

The regression tree approach and state-based approach are much alike. For both methods, first the relevant data subset for a prediction needs to be acquired (by finding the right state, or by getting the proper leaf node), after which the obtained data is used to form a prediction. However, there are two main differences between the methods.

First, for the state-based prediction method, the structure of the state tree in terms of the order of the splits is defined in advance, while for the regression tree approach the tree is fully created dynamically dependent on the data input. Because of this, the state-based method is likely to generate predictions faster than the regression tree approach and therefore is more suited for use within the Power TAC game, in which a time constraint of five seconds for each timeslot is present [19].

Secondly, when the relevant data subset is acquired, it is used in a different way to perform a prediction. In the state-based approach, we take into account the distributions for different hours and consider the relation of the most recent data compared to the data distributions. For the regression tree approach, simply the average of the outcome variables in the acquired data subset is taken. Thus, within the regression tree approach the trend of data is not directly taken into account. Due to their similarities yet also differences, it is interesting to investigate how the two prediction methods fare against each other in terms of prediction performance and time performance.

5.2 Evaluation setup

Before evaluation of both the state-based prediction method described in Section 4.3 and the regression tree approach from Section 5.1 we first construct data by using the data generation model discussed in Section 4.2. We build data for a number of customers and define multiple tariffs that are used by these customers. Based on the set of customers and tariffs we define multiple scenarios. Each scenario represents a competition of 336 timeslots in which certain customers participate and are attracted by a broker in varying ways. The customers that are part of our scenarios, are presented in Table 5.1.

The consumption for the last three customers listed in Table 5.1 is generated by the model described in Section 4.2. The consumption of *Village 1* and *Village 2* is extracted from a Power TAC default competition. For the self-generated

Table 5.1: The characteristics of the customers that are used in our data set.

Name	Type	Consuming / producing	Population
Village 1	Household	Consuming	8
Village 2	Household	Consuming	8
HouseholdConsumer	Household	Consuming	10,000
OfficeConsumer	Office	Consuming	100
FactoryProducer	Factory	Producing	50

customers we have used the following random factors and base values to generate their consumption values:

Table 5.2: The base values and the random factors of the self-generated customers.

Customer	Base value	Random factor
HouseholdConsumer	10,000	8
OfficeConsumer	10,000	15
FactoryProducer	14,000	38

The tariffs that are used by a customer are determined by using tariff setups. A tariff setup represents the way of tariff usage during a simulated competition. For example, one setup might be that a customer uses one tariff during a complete competition. Each customer that takes part in a scenario has a tariff setup assigned to it. Before we present the tariff setups that we use, we introduce the different tariffs:

- **Tariff1:** containing the rates a rate of 0.15 for night hours and a rate of 0.18 for day hours.
- **Tariff2:** containing the rates a rate of 0.12 for night hours and a rate of 0.21 for day hours.

By using one or more of the above tariffs we define multiple tariff setups:

- **Tariff setup 1 (T1):** the full population of a customer uses one tariff over the course of 336 timeslots. This tariff is *Tariff1*. This is a simple setup that allows for evaluating the performance of customer consumption on a more basic level.
- **Tariff setup 2 (T2):** the full population of a customer uses tariff *Tariff1* for the first 168 timeslots. Then, the full customer population changes

the used tariff to *Tariff2*. This a interesting setup to evaluate, since we can compare this setup to *T1* to get insight into how a sudden tariff change influences the prediction performance of the two evaluated prediction methods.

- **Tariff setup 3 (T3):** the full population of a customer uses *Tariff1* for 112 timeslots. Then, for 112 timeslots, that customer is not attracted by our broker and no tariff transactions from the customer are received. For the last 112 timeslots the customer is attracted again and uses *Tariff1*. With this setup, we can investigate how a period with a lack of data would influence the prediction results.

- **Tariff setup 4 (T4):** the full population of a customer uses a tariff over the course of 336 timeslots that is not used by any other customer. The tariff that is used contains the same rates as *Tariff1*.

We will evaluate the state-based prediction method and the regression tree forecasting model through a variety of scenarios. In each scenario certain customers are participating, and all of these customers have a specific tariff setup assigned to them. The list of evaluation scenarios and the customers that participate in them can be viewed in Table 5.3.

Table 5.3: The list of evaluation scenarios and their setups.

Scenario	Village 1	Village 2	HouseholdConsumer	OfficeConsumer	FactoryProducer
1.1	**T1**	**T1**	-	-	-
2.1	-	-	**T1**	-	-
2.2	-	-	**T2**	-	-
2.3	-	-	**T3**	-	-
2.4	-	-	**T1***	-	-
3.1	-	-	-	**T1**	-
4.1	-	-	-	-	**T1**
5.1	**T4**	**T4**	**T4**	**T4**	**T4**

In Table 5.3, when for a certain scenario it is stated that a customer uses *T1*, then that customer uses that tariff setup with a consumption redistribution percentage of 0%. When the tariff setup is notated as *T1**, then the customer associated to that tariff setup has a consumption redistribution percentage of 10%. This means that that particular customer will redistribute 10% of a day's consumption over the 24 hours of that day by taking into account the tariff rates used during those hours, as is also described in Section 4.2.

Scenario 1.1 Scenario 1.1 is an interesting one, since the customers involved in this scenario (*Village 1* and *Village 2*) and their consumption are directly obtained from actual Power TAC simulation data. For this scenario, the customers simply use one tariff for the complete duration of the simulated competition. The reasoning behind this is that both *Village 1* and *Village 2* for the most part also only used one tariff within the Power TAC pilot competition as well.

Scenario 2.2 The more interesting tariff setups are examined in conjunction with the *HouseholdConsumer* customer. In Scenario 2.1 the customer only uses one tariff, but for 2.2 to 2.4 other tariff setups are applied. In Scenario 2.2 the consumer switches tariff halfway through the competition. For the state-based method, this means that a new tariff-specific state will be used for forecasting. Since this state will initially start with a small pool of data, the predictions will have a low reliability, and thus it is to be expected that the prediction performance will also suffer in comparison to Scenario 2.1.

Scenario 2.3 In Scenario 2.3 for a total of 112 timeslots no tariffs are used by any customer. Before and after this period a customer uses *Tariff1* for 112 timeslots. By comparing the results of Scenario 2.3 with the ones from Scenario 2.1, we can investigate whether or not the prediction performance of the state-based and the regression prediction methods are negatively influenced by the temporary lack of data input. Our expectations are that both methods will have a decreased performance in comparison to Scenario 2.1.

Scenario 2.4 In Scenario 2.4 we increase the redistribution percentage from 0 to 10%. Due to this change, the customer will adjust its consumption based on the tariff it uses. This scenario can give us insight into how well the forecasting methods are able to cope with customer behaviour that is influenced by tariff rates. It is likely that the performance will be comparable to Scenario 2.1, since basically only the consumption pattern changes, while the randomness remains constant.

Scenario 3.1 and 4.1 Scenario 3.1 and 4.1 have a setup that is similar to the one of Scenario 2.1. The only difference between the scenarios is that a different customer participates within them. For Scenario 2.1 a household consumer with a relatively low random factor in its consumption pattern participates in the scenario, while for Scenario 3.1 and 4.1 respectively an office consumer and a factory producer that both have a larger random factor take part in the scenarios. By comparing to each other the prediction performance of Scenario 2.1, 3.1, and 4.1, we can investigate the level to which the state-based method and the regression tree model are affected by more noise in the data. It can be expected that the results will become worse the larger the random factor is, but it is interesting to see to which degree this will be the case.

Scenario 5.1 Last, in Scenario 5.1 we add the five different customers from the other scenarios to one single scenario and let each of the customers use a different tariff. This scenario is more representative of an actual competition scenario in which a group of customers participates. Since for the state-based prediction method the forecasts associated to one customer will not be affected by the consumption of other customers, the obtained prediction performance for the state-based method will likely not be surprising given the results of Scenario 2.1, 3.1, and 4.1. However, for the regression tree method the predictions per individual customer áre influenced by consumption of other customers, since customers, customer types and power types have an impact on how the tree is generated, and as a result the predictions for a specific customer retrieved from the built tree will be affected.

In the pilot version of Power TAC from July 2011, customer tariff changes were published only one timeslot in advance. We will evaluate the prediction methods with the assumption of perfect information regarding the attracted customers and the tariffs that they use for up to 23 timeslots into the future. This is a realistic consumption, as in reality brokers are informed of tariff adjustments much more than one hour in advance. We use this assumption to more accurately evaluate the prediction performance of our forecasting algorithm. Let us explain with an example how skewed the evaluation of prediction performance can become when evaluating without perfect information.

In timeslot z_c a prediction is required for timeslot z_{c+g}. For this timeslot, it is expected that 10 individuals associated to customer c are attracted. With this amount in mind, two predictions are formed with two different prediction methods. They are as follows:

- **Prediction method 1:** 200 kWh
- **Prediction method 2:** 110 kWh

Timeslots later the actual consumption for timeslot z_{c+g}. It appears that in between timeslot z_c and timeslot z_{c+g} five individuals related to customer c stopped being attracted by the broker. The actual consumption for timeslot z_{c+g} for these five individuals is 100 kWh. If we now examine the prediction performance of the two methods for this particular example, we see that method 1 has an error of 100, while method 2 has an error of 10. Method 2 thus is considered better. However, this does not give the right picture, because the predictions in timeslot z_c where made with 10 individuals in mind. If the predictions would be performed with the knowledge that only five individuals would be attracted in timeslot z_{c+g}, then prediction method 1 would be better.

One possibility is to construct a prediction model to forecast the customers and the tariffs that they use for future timeslots. This however requires input from other models that are part of a full Power TAC broker framework, such as a model that specifically keeps track of a broker's tariffs in relation to the tariff market. Since we focus purely on the performance of a short-term prediction method which only forms a small fraction of the total framework of a Power TAC broker agent, we evaluate with perfect information regarding the customers and their used tariffs for up to 23 timeslots into the future.

The state-based prediction method and the regression tree forecasting model will be evaluated by computing the median absolute percentage error (MdAPE) for the varying scenarios. We decided to use this measure rather than the mean absolute percentage error (MAPE), because it is more robust and less sensitive to outliers. In theory, one prediction error of infinity causes the MAPE to become equal to infinity as well. This is not the case for MdAPE, as simply the middle value of an ordered list of absolute percentage errors is taken, which does not change when there is a high prediction error outlier. For this reason, MdAPE gives a better representation of the expected absolute percentage error. The MdAPE has the disadvantage that it cannot properly handle actual values of zero, as that would result in a division by zero. However, in our evaluation scenarios no actual zero values occur except for the part of Scenario 2.3 in which no transactions are received. For this particular scenario, we only evaluate the performance for the 224 timeslots in which transactions are received. Evaluating the timeslots in which no transactions occur would also not be interesting, since our assumption of perfect information would always result in a prediction error of 0% for these timeslots. Thus, because we are not affected by zero values in our evaluation, MdAPE is a proper metric to apply. Next to using the median, we also measure the first and third quartile (defined as respectively Q_1APE and Q_3APE) of the list of absolute percentage errors. This allows us to get a view on the stability of predictions as well. When we display results of our evaluation, it will be notated in the form: (Q_1APE, MdAPE, Q_3APE).

5.3 Optimizing the state-based prediction model

In this section we will perform an evaluation analysis on the state-based prediction method by reviewing its parameter settings and by examining the prediction performance obtained under certain settings for the variety of scenarios defined in Section 5.2. The prediction method contains three different parameters.

One decision that needs to be made by the user is the tariff dissimilarity threshold to determine whether a tariff is similar enough to another and can be taken into account for predictions not associated to the tariff. This parameter should be kept low. If this parameter would be set at a high number, then two very different tariffs could be considered similar enough and consumption related to an unrepresentative tariff could be used for predictions, resulting in a bad prediction performance. The setting that we use is 5%. With this setting, only tariffs that actually are similar will be considered similar.

Another choice that needs to be made is the number of consumption values

η associated to the hour of the timeslot for which a prediction is performed that have to be stored within in a state in order to judge that state as to having enough data for a prediction. For our evaluation, we have set this parameter on 4. We consider an η of 1 to be too low and too unreliable to base predictions on. If this one value is an outlier, then the prediction would become equal to this outlier and this would result in a high prediction error. At the start of a Power TAC competition, each broker agent receives for each customer the customer's consumption of the 14 days prior to the competition's start. Within these 14 days, two weekends that place and thus each state that represents weekend consumption will have four consumption values stored for each hour. If we would set η to a value larger than 4, then the states related to weekend consumption will be considered as to not having a feasible amount of data. For this reason, we have decided to use an η of 4.

Last, there is the number of timeslots m that are taken into account to determine the timeslot trend. The higher this number, the less impact more recent data has on the eventual prediction. This can have both negative and positive consequences depending on the data that is used as input. A low number has the advantage that the method can quickly adapt based on the most recent data, but it also has the possible consequence that an outlier can worsen your results. A higher number makes the algorithm learn slowly, but less sensitive to an outlier. To determine m, we evaluate the prediction performance related to m-parameters ranging from 1 until 7 in the evaluation scenarios presented in Section 5.2.

First, we obtain the results for four basic scenarios: Scenario 1.1, 2.1, 3.1, and 4.1. In these four scenarios, only one customer is involved and only one tariff is used. These scenarios give us insight into how well the state-based method is able to perform under simple circumstances in which no difficulties occur in terms of extra tariffs. Per scenario and for each m, we compare the acquired MdAPE for that m to the average MdAPE of a set of m-values. The value that we compute for each m given the prediction gap g is as follows:

$$d_g^m = \frac{\bar{y}_g^M - y_g^m}{\bar{y}_g^M} \cdot 100, \text{ where}$$
$$\bar{y}_g^M = \frac{\sum_{i \in M} y_g^i}{|M|}$$

where M is the set of m-parameters, which in our case is $\{1,2,3,4,5,6,7\}$, and y_g^i is the MdAPE for the scenario associated to $m = i$ and prediction gap g. The MdAPE values related to all evaluation scenario for the state-based method are listed in Section C.1. By computing the percentual decrease d_g^m, we can get insight into which m values perform better for the different scenarios and also to which degree they perform better.

The results for Scenario 1.1 are displayed in Table 5.4.

Table 5.4: The percentual decrease in MdAPE for each m in comparison with the average MdAPE related to $m = \{1,2,3,4,5,6,7\}$ for different prediction gaps for Scenario 1.1.

Scenario 1.1

					m			
Gap	Avg	1	2	3	4	5	6	7
1	11.46	0.46	0.55	**2.82**	0.72	**-1.46**	**-1.72**	**-1.37**
2	12.14	**1.65**	**1.89**	-0.91	**-1.32**	0.58	**-2.47**	0.58
3	12.30	0.37	**-1.50**	-0.36	**1.19**	-0.03	**0.70**	-0.36
6	13.09	**-8.33**	-3.67	-2.14	0.69	3.13	**4.81**	**5.50**
12	12.36	**-6.24**	**-4.06**	-2.44	0.15	1.36	**4.03**	**7.19**
23	12.09	**-6.58**	0.94	0.53	1.77	1.28	0.37	1.69
Avg		-3.11	-0.97	-0.42	0.53	0.81	0.95	2.20

In Table 5.4 the results associated to different combinations of m and size of prediction gaps are summarized. When the prediction performance related to a particular m and a specific gap is a negative value, that means that the prediction error associated to that m is larger than the average prediction error of $m = \{1,2,3,4,5,6,7\}$ related to that particular prediction gap. If it is a positive value, then the prediction performance is better than the average of the seven m-values. What can be observed from Table 5.4 is that there is no m value for which the prediction performance never is below average. Furthermore, the values are not large, which means that the difference in performance is close for the different m-values. Only for $m = 1$ and a gap of 6 or larger the prediction performance is more than 6% less than average. We have bolded the results that are better or worse than the results associated to the other m-values on a significant level of 95%. As can been observed from the table, the majority of the displayed values are not bolded and thus are not significantly better or worse.

The fact that the performance between different values of m is similar can be attributed to the transformation that takes place within the state-based method through the simple linear regression model. Due to this transformation process, the timeslot trend that is acquired for predictions does not deviate much when different values of m are used, since transforming tends to move the timeslot trend closer to more average values regardless of the selected m. Furthermore, even varying timeslot trends may result in similar predictions, since the timeslot trend only represent the relative magnitude of future consumption, and if consumption is relatively stable during specific hours in terms of absolute consumption, then similar predictions will be formed no matter the acquired timeslot trend.

However, for larger values of m the obtained timeslot trends have a smaller variance and are more likely to be closer to a relatively average level, since a weighted average of more values is computed and generally smoothes to such a relatively average level. This higher level of smoothing for larger m-values can be both good and bad. Because of the smoothing the predicted timeslot trends are typically average and relatively small or large trends are not forecasted as frequently as is the case for smaller m-values. This may also explain why the prediction performance for larger m-values is better than average for timeslots that lie further ahead in the future (the ones that are linked to larger prediction gaps). These predictions are less certain, since the relationship typically weakens when timeslots are more distant from each other. Overall, by using a less risky approach through the use of larger m-values a small increase in prediction performance is gained for these distant timeslots. However, this increase is not large due to the simple linear regression transformation process that is used for determining timeslot trends and due to the timeslot trend not always having a large influence on acquired predictions, as described previously.

A value of 7 yields the best performance for forecasting with large prediction gaps, but performs somewhat less than average for smaller gaps. Overall, a value of 7 is the best choice for Scenario 1.1. Applying a value of 7 results in a prediction performance that is 2.20% better than the average performance for Scenario 1.1. This performance is not reached by using any of the other m-values. However, this is mainly due to the relatively good prediction performance for a gap of 6 or 12.

In the same way as for Scenario 1.1, Table 5.5 shows acquired results for Scenario 2.1.

Table 5.5: The percentual decrease in MdApe for each m in comparison with the average MdApe related to $m = \{1, 2, 3, 4, 5, 6, 7\}$ for different prediction gap magnitudes.

Scenario 2.1

					m			
Gap	Avg	1	2	3	4	5	6	7
1	1.38	-0.83	**2.80**	<u>**2.80**</u>	-0.10	**-1.55**	-0.10	**-3.01**
2	1.33	**-5.15**	<u>1.61</u>	1.61	0.86	1.61	0.11	-0.64
3	1.40	**-4.61**	**-2.46**	**-2.46**	1.13	**2.56**	**2.56**	<u>**3.28**</u>
6	1.36	**-4.00**	**-4.74**	-1.79	1.90	**2.63**	**2.63**	<u>**3.37**</u>
12	1.38	**3.42**	1.97	<u>**4.15**</u>	-1.66	**-2.39**	-1.66	**-3.84**
23	1.54	-1.58	**-2.88**	**-2.88**	0.37	1.67	**2.33**	<u>**2.98**</u>
Avg		-2.12	-0.62	0.24	0.41	0.76	<u>0.98</u>	0.36

Similar to the results for Scenario 1.1, Table 5.5 shows that an m-value of 1 is the least preferred choice, and again there is no setting that performs consistently better than average. Interestingly, for a gap of 3, 6 or 23 a larger value of m seems to perform better, while for a gap of 12, smaller m-values yield better results. This is most likely because due to the transformation process the actual predictions for different m-values overall do not differ much and may sometimes be better and at other times be worse, and as a consequence the results are spread and no best m-parameter value exists. Overall, a value of 6 yields the best performance even though it only performs 0.98% better than average.

To get a better overview of the overall performance for the different m-values across multiple scenarios, we have summarized the average of the results for Scenario 1.1, 2.1, 3.1 and 4.1. They are listed in Table 5.6.

Table 5.6 shows that a lower value for m is slightly preferable for predictions

Table 5.6: The average percentual decrease in MdApe for each m in comparison with the average MdApe related to $m = \{1, 2, 3, 4, 5, 6, 7\}$ for Scenario 1.1, 2.1, 3.1, and 4.1 for different prediction gaps.

Basic scenarios

	m						
Gap	1	2	3	4	5	6	7
1	-0.64	**1.29**	**1.25**	0.48	**-0.75**	-0.62	-1.01
2	**-2.81**	**2.11**	0.85	-0.66	0.26	0.27	-0.03
3	**-2.48**	-1.14	0.07	**1.65**	**2.05**	0.73	-0.87
6	**-4.15**	**-2.28**	-1.25	0.90	1.91	**2.28**	**2.59**
12	0.21	-0.54	**1.03**	-0.56	**-1.08**	-0.06	**1.00**
23	**-3.36**	-0.30	-0.17	0.56	**1.48**	0.86	0.93
Avg	-2.20	-0.14	0.30	0.39	0.64	0.58	0.44

related to a smaller prediction gap, while higher values of m are preferred for forecasts with a larger gap. A value of 1 does not seem to be a good choice, since the obtained results that are associated to this value usually are below average and most often have the worst prediction performance of all m. Also, using an m-value of 2 generates less than average results for the larger gaps.

In general, the results do not seem to deviate much between different values of m, and there definitely is no perfect m-value for the data that is evaluated. The overall best choice of $m = 5$ only performs 0.64% better than average. A conclusion we can draw in terms of m-value selection is that a value larger than 1 should be selected, since picking a value of 1 most consistently yields less than average prediction performance results. Furthermore, overall a larger value for m results in slightly better prediction performance for forecasts related to larger prediction gaps. However, it should be pointed out that this is not the case for every scenario. Smaller m-values have the advantage that less computations are required to form predictions, as less historical data is used to compute them. The fact that the results between different values of m are so similar and the best choice between scenarios shift from one m to another can be attributed to the simple linear regression model used within the state-based prediction method.

All in all, there simply is no preferred choice for m. In order to evaluate our state-based prediction method with the regression tree method we therefore select and evaluate multiple m values, ranging from $m = 2$ to $m = 7$. By evaluating based on several m values we can get a more broad overview of the performance of the state-based method in comparison with the regression tree approach than if we would evaluate based on one specific m-value. Instead of reporting the results related to one specific m-value, we will therefore show per scenario averaged prediction results associated with $m = \{2, 3, 4, 5, 6, 7\}$, as well as the worst and best prediction results obtained by this set of m-values.

5.4 A comparison of prediction methods

In this section we compare the state-based prediction method to the regression tree forecasting approach described in Section 5.1. The comparison of the two methods will be done on two different levels. First the prediction performance will be analysed. Then, the time performance of both prediction methods will be illustrated.

5.4.1 Prediction performance

In Section 5.1 we described our used regression tree approach. We apply this approach to the scenarios listed in Table 5.3. The results for the regression tree method are depicted in Section C.2. Since the regression tree approach contains a random component that determines which data is used as training data, the predictions that are computed by this method will deviate between multiple runs. We decided to report the results for each scenario based on the average of obtained results for ten runs. Added to the results is also the lower limit and the upper limit based on a 95% confidence level. This gives insight into the deviation of the prediction performance between different runs of the regression tree model. It also allows us to determine whether the state-based method is significantly better (i.e. has a prediction error below the lower limit) or significantly worse (i.e. has a prediction error larger than the upper limit) than the regression tree prediction method for the different scenarios.

As described in Section 5.3, no optimal m-parameter exists and evaluating the prediction performance of the state-based method using multiple m-parameter values could provide us a broader insight into the performance in relation to the regression tree method. For $m = \{2, 3, 4, 5, 6, 7\}$, we have gathered the results and compare for each scenario and several prediction gaps the average, the worst, and the best obtained results with the acquired results for the regression tree model. In Table 5.7 the percentual decrease of results for the state-based method in relation to the average obtained result for the regression tree method are summarized. When the decrease is significant, the value is bolded.

Table 5.7: The percentual decrease in MdAPE obtained by the state-based method in comparison with the average MdAPE of the regression tree approach for all scenarios.

		Gap					
Sc.		1	2	3	6	12	23
1.1	Worst	**14.58**	**7.51**	**7.69**	0.44	**7.55**	**14.90**
	Avg	**15.96**	**9.49**	**9.00**	**5.29**	**12.08**	**15.52**
	Best	**18.39**	**11.45**	**10.13**	**9.24**	**17.54**	**16.10**
2.1	Worst	**6.58**	**11.26**	**3.38**	**4.05**	**4.03**	**-4.64**
	Avg	**9.43**	**12.58**	**6.42**	**9.01**	**7.05**	-1.43
	Best	**11.84**	**13.25**	**8.78**	**11.49**	**11.41**	1.32
2.2	Worst	**9.33**	**8.50**	**3.92**	**8.33**	**2.65**	0.00
	Avg	**10.33**	**10.89**	**6.54**	**11.00**	**4.97**	2.58
	Best	**12.00**	**13.07**	**7.19**	**14.10**	**7.28**	**5.81**
2.3	Worst	-0.69	-0.70	-0.70	**-3.45**	4.61	**2.61**
	Avg	0.46	**3.40**	**4.55**	**5.98**	**6.80**	**7.73**
	Best	2.08	**5.63**	**7.69**	**9.66**	**7.89**	**10.46**
2.4	Worst	-2.00	**-5.96**	**-8.05**	**-6.71**	**-9.27**	-1.86
	Avg	1.56	-1.21	**-6.49**	**-5.03**	**-8.39**	**4.66**
	Best	**4.00**	**3.97**	**-4.70**	**-2.68**	**-7.28**	**6.83**
3.1	Worst	**4.33**	**2.80**	-2.17	**-4.62**	**-7.10**	**-2.81**
	Avg	**5.78**	**3.88**	1.40	**-3.54**	**-5.50**	-0.57
	Best	**7.43**	**5.28**	**4.04**	-2.15	**-4.01**	1.88
4.1	Worst	**2.77**	0.30	**-5.64**	**-10.25**	**-7.56**	**-10.19**
	Avg	**4.17**	**3.01**	-1.83	**-8.53**	**-6.31**	**-8.40**
	Best	**5.30**	**7.06**	0.91	**-6.57**	**-4.59**	**-6.79**
5.1	Worst	**17.52**	**16.31**	**17.32**	**13.64**	**9.55**	**11.71**
	Avg	**20.01**	**19.31**	**18.76**	**14.17**	**13.41**	**15.54**
	Best	**23.08**	**21.46**	**20.78**	**15.00**	**15.91**	**18.02**

The results from Table 5.7 show us that for Scenario 1.1 the state-based method performs better than the regression tree approach. The decrease in MdAPE reaches to a maximum of 18.39%. The magnitude of the prediction gap does not seem to have an impact on the performance of the state-based method in relation to the regression tree model, since the performance improvement is still 17.54% and 16.10% for respectively a gap of 12 and one of 23.

For Scenario 2.1 the state-based method performs better than the regression tree approach. For all predictions gaps except 23 the worst obtained prediction performance for the state-based method is still significantly better than the performance of the regression tree model. For Scenario 3.1 and 4.1, which are similar to Scenario 2.1, the results mostly favour the regression tree method. For prediction gaps of 1 and 2 the state-based method has an improved performance, but for gaps of at least 6 the regression tree performs better. The difference between Scenario 3.1 and 4.1, and Scenario 2.1 is that the customers involved in Scenario 3.1 and 4.1 have more randomness in their consumption (of which the largest randomness appears in 4.1). Based on the obtained prediction results, a regression tree seems to be better able to withstand noise and seems more robust.

For Scenario 2.1, 3.1, and 4.1 the performance of the state-based method relative to the performance of the regression tree approach roughly seems to decrease as the prediction gap increases. This means that the regression tree approach is more capable of forming prediction for more distant timeslots.

In Scenario 2.2, in comparison to Scenario 2.1, we introduced a tariff change halfway through the competition. This does not seem to negatively affect the performance of the state-based method in comparison with the regression tree approach. In Table 5.8 one can observe the absolute as well as the relative decrease in MdAPE for Scenario 2.2 in relation to Scenario 2.1 for both the state-based and the regression tree prediction method.

Table 5.8: The average absolute and percentual decrease in MdAPE obtained by the state-based method in comparison with the absolute and percentual decrease in MdAPE of the regression tree approach for Scenario 2.2 in relation to Scenario 2.1.

		Gap					
Method	**Decrease**	1	2	3	6	12	23
State-based	Absolute	0.03	-0.04	-0.04	-0.04	-0.05	0.02
	Relative (%)	2.27	-3.28	-3.28	-3.12	-3.64	1.40
Regr. tree	Absolute	0.02	-0.02	-0.03	-0.06	-0.02	-0.01
	Relative (%)	1.31	-1.66	-2.35	-4.15	-1.41	-0.78

The results from Table 5.8 teach us that both the state-based method and the regression tree approach mostly perform worse when a tariff change occurs for a customer halfway through the competition. This result is not surprising, since the tariff change causes for both prediction methods a shift to a newly used data pool which contains less values, and thus the forecasts for both methods initially become less reliable after the tariff change. It can be expected that when more tariff changes happen, the performances drop further for both methods. The degree to which there is an increase in MdAPE on average is similar for both methods.

For Scenario 2.3 the involved customer does not use a tariff for the middle third of the competition. This allows us to examine to which degree a period in which no consumption data is acquired affects the prediction performance of the two prediction methods. For Scenario 2.3 we have compared the prediction performance in the period after the temporary lack of data. The absolute and the relative decrease in MdAPE for the last 112 timeslots for Scenario 2.3 relative to Scenario 2.1 are displayed in Table 5.9 for both the state-based method and the regression tree approach.

Table 5.9: The average absolute and percentual decrease in MdAPE obtained by the state-based method in comparison with absolute and percentual decrease in MdAPE of the regression tree approach for Scenario 2.3 in relation to Scenario 2.1 for the last 112 timeslots.

		Gap					
Method	**Decrease**	1	2	3	6	12	23
State-based	Absolute	0.04	0.08	0.20	-0.06	0.08	0.26
	Relative (%)	2.34	5.08	11.57	-3.56	4.58	13.80
Regr. tree	Absolute	0.11	0.07	0.01	-0.04	-0.04	0.17
	Relative (%)	7.12	4.70	0.77	-2.95	-2.49	10.63

Table 5.9 shows that for both prediction methods the MdAPE of the last 112 timeslots of Scenario 2.3 for the most part surprisingly decreases in comparison to the last 112 timeslots of Scenario 2.1. This means that a period without data does not necessarily negatively influence the prediction results.

In Scenario 2.4 the customer from Scenario 2.1 shifts its consumption based on the tariff that it uses. On hours where the tariff rate is relatively low, the customer will consume more, and vice versa. For this scenario the customer shifts a total of 10% of its daily consumption over the hours of that day. In Table 5.10 the absolute as well as the relative decrease in MdAPE relative to

Scenario 2.1 for both the state-based and the regression tree prediction method are listed.

Table 5.10: The average absolute and percentual decrease in MdAPE obtained by the state-based method in comparison with absolute and percentual decrease in MdAPE of the regression tree approach for Scenario 2.4 in relation to Scenario 2.1.

		Gap					
Method	**Decrease**	1	2	3	6	12	23
State-based	Absolute	-0.10	-0.21	-0.20	-0.22	-0.25	0.00
	Relative (%)	-7.27	-15.77	-14.65	-16.32	-18.26	-0.21
Regr. tree	Absolute	0.02	0.00	0.00	0.01	-0.02	-0.07
	Relative (%)	1.12	0.13	-0.07	0.60	-1.41	-4.89

The results in Table 5.10 clearly show that for Scenario 2.4 the results are much more negatively influenced for the state-based method than for the regression tree approach. The main reason for the increase in MdAPE in comparison with Scenario 2.1 for the state-based method is that the historical data that is used at the beginning of a competition to initialize states with is less representative for the consumption data during the actual game for Scenario 2.4 than for Scenario 2.1. The acquired historical data of a customer represents consumption without a specific tariff in mind, and thus that consumption simply represents a pattern that is not affected by specific tariff rates. During the actual competition in Scenario 2.4, the customer adjusts its consumption based on the tariff that it uses and the consumption becomes significantly different from the historical consumption. This is not the case within Scenario 2.1, since in that scenario the customer does not change its consumption based on tariff rates. At the start of the competition the historical data will be used to form predictions for the state-based method, since there is not enough sufficient tariff specific data available. As a consequence, the not so representative historical data will be used at the start of Scenario 2.4 and the prediction errors will be larger at the beginning of Scenario 2.4 than at the start of Scenario 2.1, resulting in a larger MdAPE.

In Section 5.3 we explained that in our parameter settings a state should contain at least four consumption values to consider the state as to having a feasible amount of data. For Scenario 2.4 a value smaller than four would have a positive influence on the prediction performance. This is explained by the fact that the unrepresentative historical data is used for more timeslots the higher the minimum required amount of consumption values is, since it will take longer for the states that contain the representative data to have a sufficient amount of data.

The last scenario for which we compare the state-based method and the regression tree prediction model is Scenario 5.1. In this scenario five customers each use a different tariff and are attracted for the complete duration of the simulated competition. The results from Table 5.7 show that the state-based method performs significantly better than the regression tree approach regardless of the prediction gap. Thus, in a scenario more representative of a more active competition that involves multiple customers, the state-based method is able to perform better than the regression tree model.

In Table 5.11 the average percentual decrease in MdAPE for the state-based method relative to the MdAPE for the regression tree method for all eight scenarios are summarized.

Table 5.11: The percentual decrease in MdAPE obtained by the state-based method in comparison with the average MdAPE of the regression tree approach for all scenarios.

		Gap					
Sc.		1	2	3	6	12	23
	Worst	6.55	5.00	1.97	0.18	0.56	1.22
Avg	Avg	8.46	7.67	4.79	3.54	3.01	4.45
	Best	10.52	10.15	6.85	6.01	5.52	6.70

As can be observed in Table 5.11, for a gap of 1 or 2, over all eight scenarios the average percentual decrease in MdAPE for $m = \{2, 3, 4, 5, 6, 7\}$ for the state-based method in comparison with the regression tree approach is approximately 8%. An increase in the prediction gap results in performance declines of 4.79%, 3.54%, 3.01%, and 4.45% for prediction gaps of 3, 6, 12, and 23 respectively. An interesting observation is that the state-based method has a better performance in relation to the regression tree method for a gap of 23 than for a gap of 12. Thus, relative to the regression tree approach, the state-based method retains the same level of robustness for prediction gaps that are larger. Furthermore, the state-based method also still performs better despite of the decline in prediction performance in relation to the smaller gaps of 1 or 2.

5.4.2 Time performance

An important aspect for prediction methods that are used within the framework of a Power TAC broker agent is the required computation time. For each timeslot within a Power TAC competition a broker agent is obligated to take its actions for that timeslot within the time frame of a timeslot, which is five seconds. Since forecasting the energy imbalance of a broker agent is not the only task that needs to be performed, the computation time required for doing the imbalance predictions may not last longer than a few seconds. The computation time of the introduced state-based prediction method is compared to the computation time of the regression tree approach. We used $m = 7$, i.e. the setting which requires the most computations of the settings in Section 5.3, as the parameter setting for the state-based method. The system that we have used to perform the time evaluation experiments is as follows: Intel® Core i5-750 processor (4 CPUs, 2.67Ghz) with 4096MB RAM memory.

The computation time will be evaluated using several different setups. In each setup a Power TAC game is simulated using a specified number of tariffs n_t and a specified amount of customers n_c. The number of tariff transactions that take place within one timeslot are dependent on the specified numbers and is equal to $n_t \cdot n_c$. Each customer uses each tariff once per timeslot. Logically, the larger the number of customers and the larger the number of tariffs, the higher the amount of tariff transactions and consequently the higher the computation time required to perform all necessary predictions.

Figure 5.1 depicts the maximum encountered computation time for a single timeslot given a specified number of tariffs and amount of customers for the state-based prediction method.

In Figure 5.1 one can observe the maximum computation time encountered during simulations of 336 timeslots. Logically, the computation time becomes larger with an increase of customers and tariffs, since such an increase means that the number of transactions grows. For example, for the setup of four customers and four tariffs, the total number of transactions per timeslot is equal to 16 (four tariff transactions for each customer). The computation time

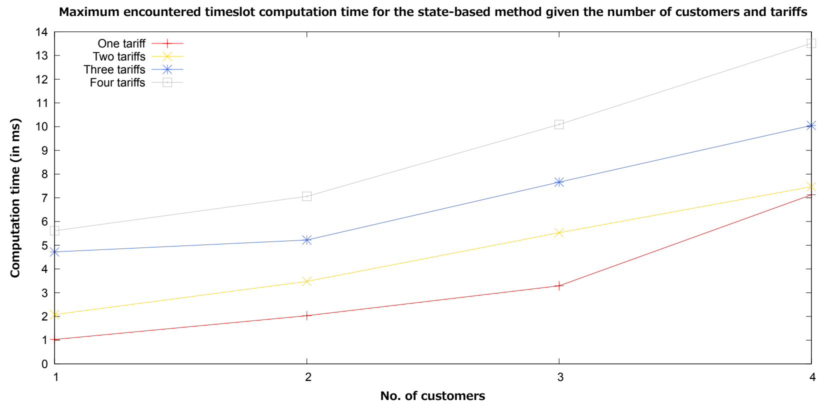

Figure 5.1: The maximum encountered computation time required for a single timeslot for the state-based prediction method given the number of customers and the amount of tariffs that they use per timeslot.

for this large setup is approximately 13.5 milliseconds. In comparison with the setup that includes only one tariff and one customer, this is an increase of approximately 1210%. Since the number of transactions has increased from 1 to 16 (an increase of 1500%) between these two setups, the computation time per transaction actually has decreased.

We have also stored the computation time values for the regression tree approach. In Figure 5.2 the maximum encountered computation time for a single timeslot given the specified number of tariffs and amount of customers for the regression tree prediction method is displayed.

Figure 5.2: The maximum encountered computation time required for a single timeslot for the regression tree prediction method given the number of customers and the amount of tariffs that they use per timeslot.

As Figure 5.2 illustrates, the computation for the regression tree method grows rapidly. In comparison with the state-based method, the computation time is much larger. Whereas the state-based method requires less than a maximum of 14 milliseconds per timeslot for the largest examined setup, the regression tree approach needs more than 8200 milliseconds.

Furthermore, we have only included one customer type and one power type in our time evaluation setups. Including more customer types or power types will increase the computation of the regression tree method even further, since the number of possible splits when building a regression tree will then become larger. The state-based method does not use the customer types and power types for forecasting, since the predictions are always formed per individual customer. Thus, for the state-based method the inclusion of more customer types or power types would not increase the computation time.

Next to the maximum encountered computation time we also investigated the computation time per timeslot over the course of 336 timeslot to observe to what degree the computation time is increasing over the course of a simulated competition. Figure 5.3 depicts for 336 timeslots the total computation time taken up until that timeslot for the setup involving four customers and four tariffs for the state-based prediction method.

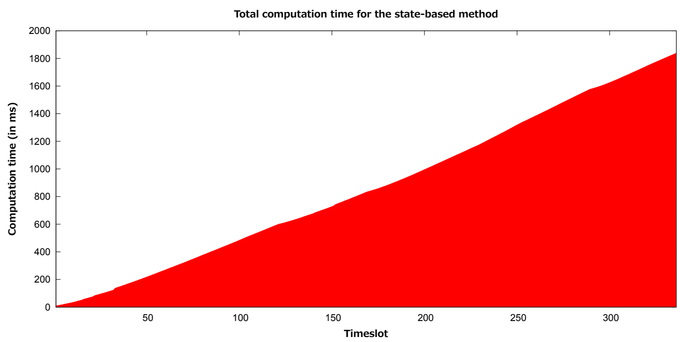

Figure 5.3: The total computation time for the state-based prediction method.

Figure 5.3 illustrates that for the state-based prediction method, the total computation time grows in a linear fashion. The computation time required per timeslot thus remains practically constant throughout the competition, otherwise the figure should have depicted an exponential increase. Thus, the state-based method is hardly affected by the number of transactions that have taken place prior to a timeslot.

In Figure 5.4 we display the total computation time over 336 timeslots for the regression tree prediction method for the same setup as for the state-based method. During each timeslot, 16 transactions take place that are distributed over four different customers and four different tariffs.

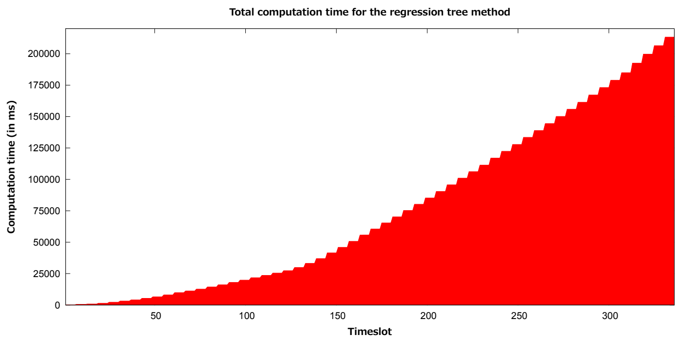

Figure 5.4: The total computation time for the regression tree prediction method.

From Figure 5.4 one can observe that for the regression tree approach the total computation time required by the regression tree method increasingly grows. The regression tree method thus is more affected by the number of past transactions that have taken place than the state-based method. This result is logical, since the further into a simulation game, the more tariff transactions are used for building and pruning a regression tree and thus the longer it takes to construct this tree. For the state-based method, no tree is built based on past tariff transactions and this considerably saves computation time.

The main conclusion we can draw from our evaluations is that overall the state-based method presented in Section 4.3 is a better predictor than the regression tree method explained in Section 5.1. In terms of prediction performance the state-based method seems to perform better overall. We found that the best prediction performance is obtained for the state-based method when the prediction gap is small. For prediction gaps of 3 or larger the performance improvement over the regression tree method declines, but still better results are acquired for the state-based method.

From our evaluation results, we also conclude that the regression tree approach is more robust than the state-based method. This is illustrated by the fact that the prediction performance is less affected by randomness in customer consumption than the state-based method. While for customers with a relative small randomness in consumption the state-based method was clearly better, evaluation results for customer consumption with a larger degree of randomness seems to turn the tide in favour of the regression tree approach. Nevertheless, the state-based method still performs better at a significant level for a prediction gap of 1 or 2 even for scenarios involving more consumption randomness. Overall the state-based method has an improved prediction performance.

Last, we found that initializing the two prediction methods with data that is not entirely representative for the actual game data, as was the case for Scenario 2.4, negatively affects the state-based method to a larger degree than the regression tree model. However, this was also caused by our parameter selection. We only use states to form predictions when they contained at least four data observations relevant to the forecast. Thus, the algorithm requires some time to use the states that contain data acquired during the game, and that do not contain the unrepresentative data acquired during the initialisation process. By lowering the selected parameter value, the states not containing the unrepresentative data would be used more quickly and representative would be applied for our forecast, resulting in improved prediction performance. However, using a low parameter setting would not be optimal when historical data actually is representative, as is the case for scenarios other than Scenario 2.4.

In terms of time performance, the state-based method performs much better than the regression tree prediction model. The computation time for the state-based method increases linearly with the number of customers and the number of used tariffs, the required time for computations for the regression tree approach increasingly grows for each customer or tariff that is added. Furthermore, the state-based method is able to perform its predictions in much shorter time spans than the regression tree approach. Our results showed that in a competition that contains four customers that have a total of sixteen transactions per timeslot by each using four tariffs, the regression tree approach requires more than 8200 milliseconds for forming its prediction for a single timeslot. The state-based method needed only approximately 14 milliseconds to acquire its prediction, only a small fraction of the time required when using the regression tree method.

Also, whereas for the state-based method the computation time per timeslot remains practically constant when more and more transactions have been stored, the computation time per timeslot for the regression tree mode clearly increases. As a result, the regression tree requires more and more time as a Power TAC competition continues and could risk crossing the Power TAC time limit of 5 seconds per timeslot. This is not the case for the state-based method. All in all, for use within a simulation game such as Power TAC where time is limited and computations need to be performed quickly, the state-based method is much more applicable than the regression tree model.

Chapter 6

Conclusions

In this chapter we conclude our research by providing a summary of our investigations. Section 6.1 presents the answers to the various questions presented in Chapter 1 through a series of statements based on our thesis work. Section 6.2 then discusses potential directions for future work regarding the state-based prediction method proposed in this thesis.

6.1 Summary and Contributions

Our main focus for this thesis was to present and evaluate a prediction method that is able to properly perform forecasts. We designed a prediction model for the Power TAC competition. In this competition so-called broker agents act on an energy market and need to perform multiple tasks, such as attracting customers by offering tariffs and trading energy on a wholesale market, with the goal of maximizing profits. One of the tasks of the agent is to resolve the expected imbalance between energy demand and supply on the short-term. In order for a broker to resolve these imbalances, first a forecast is required for determining the expected imbalance. Our aim was to build a prediction method that is capable of predicting short-term energy imbalance of future timeslots in order to serve as an initiator for resolving actions. Therefore, the main research question introduced in Chapter 1 is as follows:

- **How to predict short-term energy imbalance between supply and demand of power of a broker agent in Power TAC?**

To provide an answer to this question, we introduced a number of subquestions that each covers a different area. Through a discussion of statements related to these subquestions we form a picture of the usage of a short-term prediction method within Power TAC.

We identified several features that influence the total consumption, and created customer consumption/production data based on tariffs and daily patterns per customer type.

In Chapter 3 we discussed the general components of Power TAC and performed an analysis of the Power TAC pilot competition that took place in July 2011. Then, in Section 4.3 we described our findings on a number of features that would influence the total consumption associated with a broker agent. First, this consumption relies on the customers that are attracted by the broker. Each customer is linked to a customer type and power type which determines respectively the consumption pattern of a customer and whether the customer is consuming or producing. The consumption of a customer is dependent on the time of consumption. During weekends a customer is expected to consume in different quantities than during regular weekdays. The tariff that is used by a customer also plays a role. Brokers can control to a degree the consumption of customers by offering specific tariff rates for certain hours. In this way, brokers can encourage customers to shift some of their consumption to specific hours in order to balance the expected supply and demand of energy.

The analysis in Chapter 3 taught us that using the available Power TAC data as a source for performing our predictions would not be a wise decision, since the customer consumption data of the Power TAC competition was too limited to properly apply and test a prediction method on. For this reason, we created our own data based on the Power TAC game specification, as discussed in Section 4.2. The data constructed encompasses a number of features to ensure that customer consumption varies between different types of customers and between varying tariffs. A consumption pattern that is dependent on the customer type is applied as a base line to generate data. Customer behaviour may also be influenced by the tariffs that customers use. The main idea is that customers sacrifice a proportion of their daily consumption which is then distributed over all the hours of a day. The lower the tariff rate for a certain hour is in relation to rates of other hours, the more the customer will consume on that hour.

We developed a state-based prediction model that is able to quickly form predictions and consider customer, tariff, and time-related aspects of energy transactions in order to perform well in Power TAC.

In Chapter 2, we analysed prediction methods that have been used in the past to perform forecasts. The methods that we reviewed were exponential smoothing methods, artificial neural networks, regression models, and decision trees. Artificial neural networks, nonlinear regression models and decision trees all suffered from relatively large computation times that would likely render then inapplicable for performing predictions within the Power TAC environment, in which predictions need to be formed in a limited amount of time. Linear regression models are quicker, but are incapable of predicting complex nonlinear relationships. Exponential smoothing methods are either simplified and restricted in performing proper forecasts, such as is the case for simple exponential smoothing, or require proper parameter optimisation as initialising double exponential or triple exponential smoothing methods with the wrong parameter settings hurt the prediction performance.

In Section 4.3, we introduced our state-based prediction method. This method is able to form short-term energy predictions. In Power TAC, the algorithm generates predictions for up to 23 timeslots ahead. The procedure for creating predictions is based on using states. Each state represents a unique combination of data features. The applied features are the ones that we described in Section 4.3, such as the tariff, the customer, and the part of the week in which consumption takes place. A state includes all data that is associated to the features it represents, and based on this data predictions are formed. Unlike for decision trees where a tree is formed dynamically based on the used data, states and their relationships are defined in advance and thus are static in order to save computation time, since time is limited within the Power TAC framework and it is vital that forecasts are made timely.

The consumption prediction for a certain timeslot using a single state is dependent on the consumption for the previous timeslots and the consumption hour that is related to the prediction. By using a both a simple linear regression model as well as a weighted averaging method that assigns the highest weight to the most recent observations, we compute a timeslot trend that represents the expected relative magnitude of future energy consumption. The larger the timeslot trend, the larger the predicted consumption for a future timeslot.

The state-based method contains three parameters. One of them is the number of timeslots m taken into account to form the timeslot trend that is applied for forming predictions. The other two parameters are associated with the maximum allowed dissimilarity between tariffs in order for them to replace one another for predictions, and the number of data observations η that a state should contain that are associated with the future timeslot's hour in order for the state to be considered for using predictions. A larger m-value creates more smoothing in the weighted method applied for determining the timeslot trend. An increase in η means that predictions are based upon more observations, but also that a state is more quickly deemed as to not having enough data. When this is the case, another state that represents similar features will be used instead for prediction, and this could hurt the prediction accuracy if the data contained in the new state is not representative for the replaced state.

We designed several scenarios and conducted computer simulations to test the performance of our prediction model. The results show that the state-based prediction method performs better than the CART regression tree model.

In Chapter 5 we performed both a prediction performance and a time performance evaluation. The state-based prediction method described in Section 4.3 was compared to the CART regression tree model covered in Section 5.1. In order to do our evaluation, we introduced a total of eight evaluation scenarios in Section 5.2. Each of the scenarios represents a competition of 336 timeslots in which a Power TAC broker agent attracts a specific customer through specific tariffs. The customers vary over the different scenarios. A total of five customers were present, which contain a total of three different customer types.

Two household consumers were extracted from Power TAC default competition data, and one household consumer, one office consumer, and one factory producer were defined by ourselves and created by applying the consumption generation model described in Section 4.2. Each of the customers had a different consumption pattern, based on their customer type and a random factor. For each of the eight scenarios, we have compared the MdAPE of the two prediction methods. We analysed not only predictions that are formed one timeslot ahead, but reviewed the performance for 2, 3, 6, 12, and 23 timeslots in advance (which we refer to as the prediction gap).

Before analysing the performance of the state-based prediction method in relation to the regression tree prediction model, we examined the prediction performance of the state-based method under varying parameter settings in Section 5.3. We found that there was no definite optimal parameter choice as obtained performance results did not seem to differ significantly.

Overall the state-based prediction contained better prediction performance in our experiments, but depending on the evaluation data and the prediction gap the improvement in performance could vary greatly. In general we found that the larger the randomness in the consumption behaviour of a customer and the larger the prediction gap, the lower the performance of the state-based method in comparison with the regression tree model.

For a gap of 1 or 2, the average prediction performance over all eight scenarios related to the average performance for $m = \{2, 3, 4, 5, 6, 7\}$ for the state-based method is approximately 8% better than the regression tree method. For larger gaps this increase in prediction performance declines to approximately 4.79%, 3.54%, 3.01%, and 4.45% for respectively prediction gaps of 3, 6, 12, and 23. Interestingly, though, the increase in performance is larger for a gap of 23 than for a gap of 12. This means that the state-based method contains its level of robustness relative to the regression tree method across larger prediction gaps. Even though there is a decline in comparison with a gap of 1 or 2, the state-based method also still performs better than the CART regression model on average.

We also performed an evaluation on the time performance for both prediction methods. We have analysed the required computation time given a specific number of customers and a specific number of tariffs that are used by these customers. Based on our results, we found that the state-based method is significantly quicker than the regression tree method. The time required for performing the forecast for a single timeslot when four customers use four tariffs reached up to 8200 milliseconds for the regression tree approach, whereas the state-based method only needs up to 14 milliseconds. Furthermore, the computation time for the regression tree method increasingly grows per timeslot as the number of past transactions that have taken place becomes larger, while for the stated-based approach the required computation time per timeslot remains practically constant regardless of the number of past transactions. Thus, the state-based method contains an overall better prediction performance than the regression tree approach, as well as a significantly improved time performance. For these reasons, the state-based method is preferred over the CART regression tree model in the domain of Power TAC.

6.2 Limitations and future work

In this section we discuss some limitations of our research as well as potential directions for future work. The largest limitation is that the state-based prediction framework was only applied and evaluated separately without any integration with other Power TAC components. As a result, our possibilities were limited, as we were restricted to using pre-made data and could not utilise the algorithm during an actual dynamic Power TAC competition.

In our thesis, we made the assumption of perfect information regarding the usage of tariffs by customers for the short-term (23 timeslots ahead) in our evaluation analysis since no prediction method was included for forecasting the expected tariffs on the short-term. During an actual Power TAC competition, a broker only receives such information one timeslot in advance. If this time frame remains the same for future versions of Power TAC, a predictor is required that provides forecasts about the expected tariffs and customers for the short-term.

In the future, the prediction method could be implemented within a complete framework of a broker agent. The algorithm could then also be evaluated in a dynamic Power TAC competition instead of by utilising pre-made data.

Through other broker framework components, such as one that scouts tariffs for managing the long-term energy imbalance, more tariff-related information can be obtained and a predictor for expected customers and tariffs could be developed and incorporated with our prediction method.

The state-based method allows for customization within the methods that are applied within its framework. For this thesis we have used three characteristics to acquire the relevant states for each prediction: the part of the week, the customer, and the tariff. In the future of Power TAC, more characteristics could be added to predict energy consumption, such as features that are associated with the weather. Since the application of states is general, this part of the prediction method can also be applied in other domains by using features relevant for that domain as determinants for states. Another possible addition to the model is the inclusion of state merging for states that are similar to increase the size of data sets contained within states.

For this thesis, we have applied a weighted average method in combination with a simple linear regression model to determine a timeslot trend that represents the expected relative magnitude of future consumption. The state-based method allows for using different prediction setups within the states. One could use other weighted methods to determine the timeslot trend, or replace the simple linear regression component with another regression model, such as a segmented one.

Appendix A

Pilot broker attraction

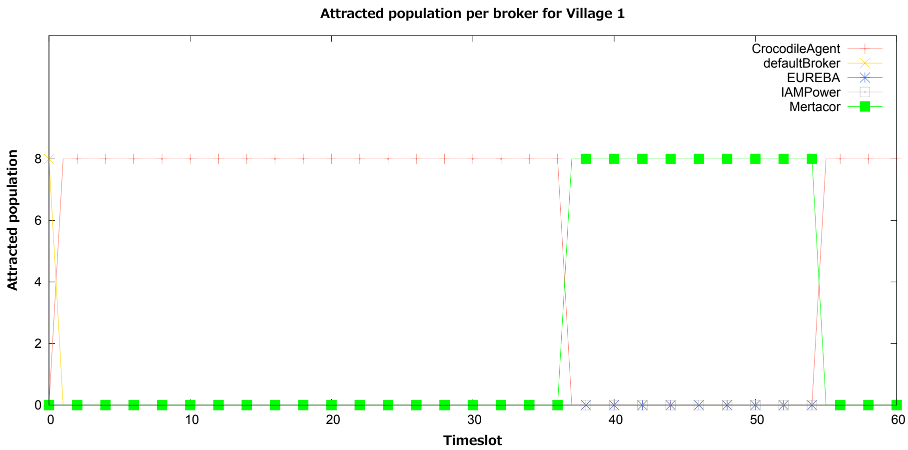

Figure A.1: The distribution of the population of the *Village 1* customer over the brokers of the pilot competition. This customer has a population of 8.

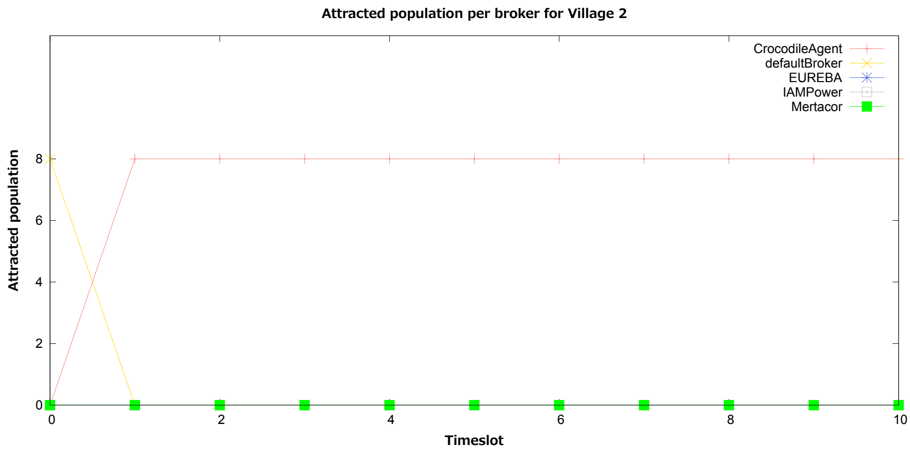

Figure A.2: The distribution of the population of the *Village 2* customer over the brokers of the pilot competition. This customer has a population of 8.

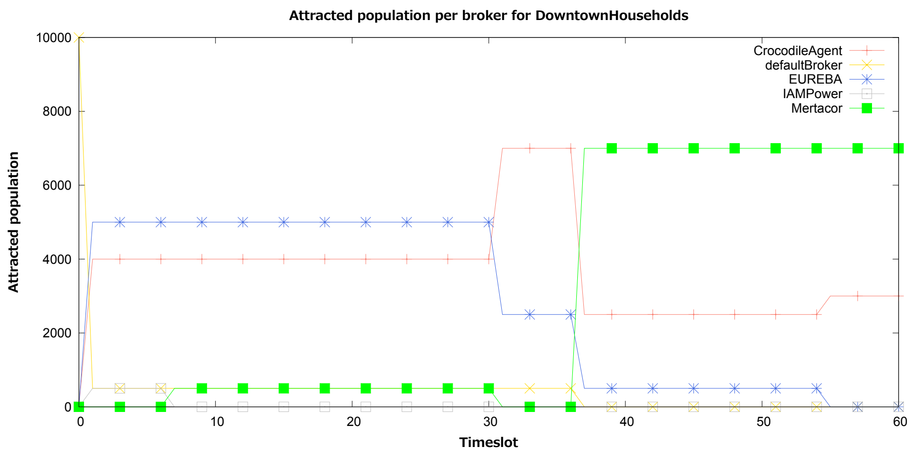

Figure A.3: The distribution of the population of the *DowntownHouseholds* customer over the brokers of the pilot competition. This customer has a population of 10,000.

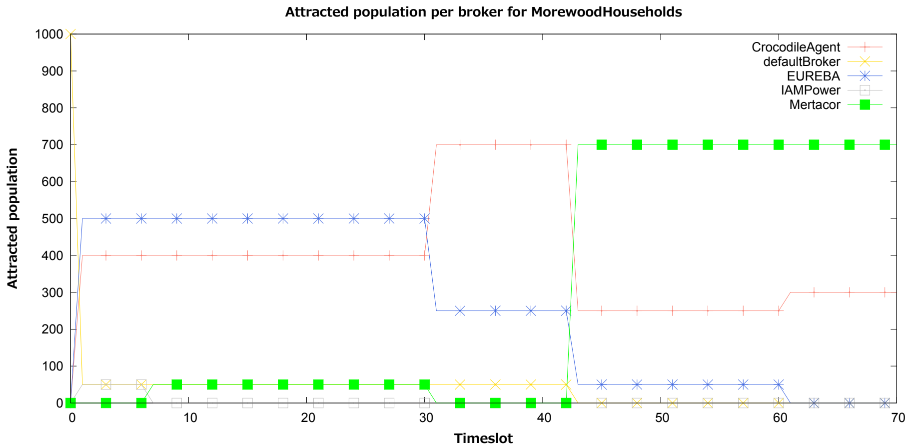

Figure A.4: The distribution of the population of the *MorewoodHouseholds* customer over the brokers of the pilot competition. This customer has a population of 1,000.

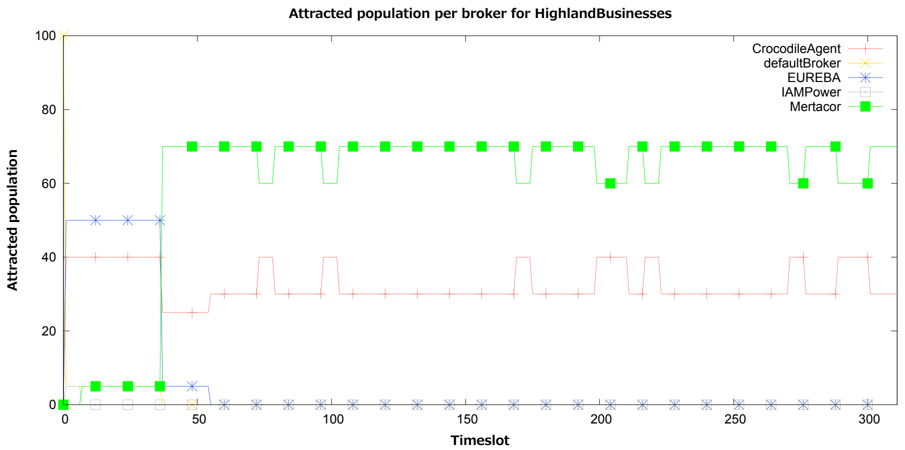

Figure A.5: The distribution of the population of the *HighlandBusinesses* customer over the brokers of the pilot competition. This customer has a population of 100.

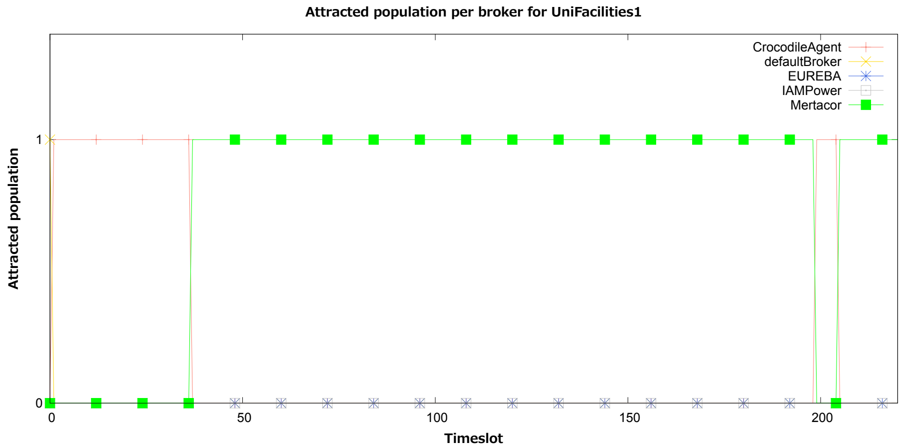

Figure A.6: The distribution of the population of the *UniFacilities1* customer over the brokers of the pilot competition. This customer has a population of 1.

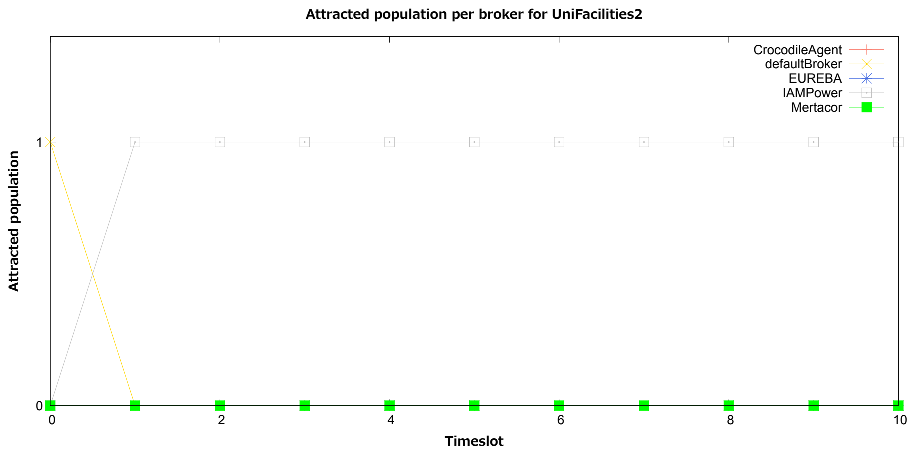

Figure A.7: The distribution of the population of the *UniFacilities2* customer over the brokers of the pilot competition. This customer has a population of 1.

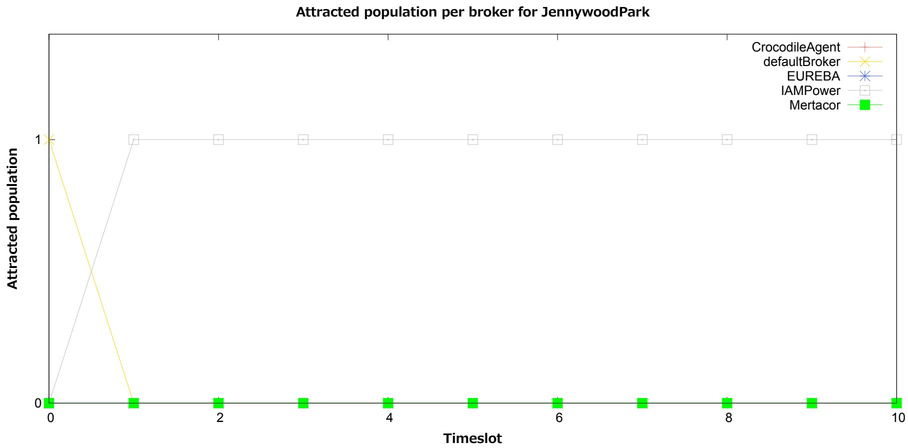

Figure A.8: The distribution of the population of the *JennywoodPark* customer over the brokers of the pilot competition. This customer has a population of 1.

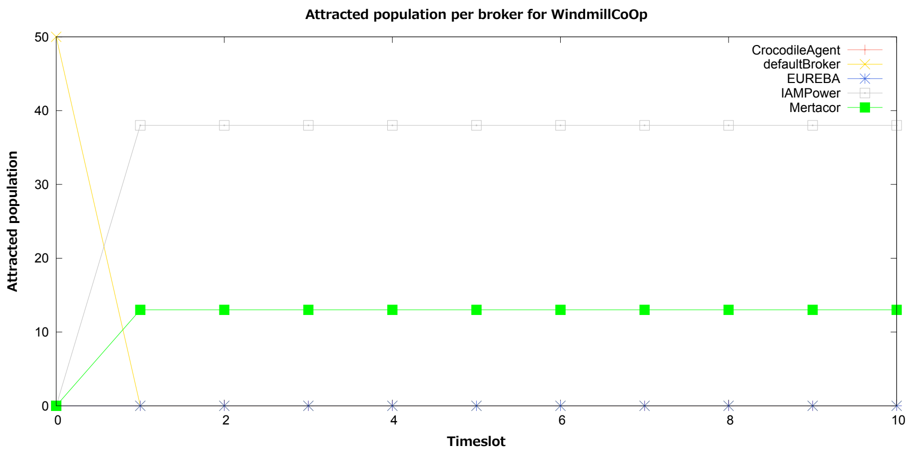

Figure A.9: The distribution of the population of the *WindmillCoOp* customer over the brokers of the pilot competition. This customer has a population of 50.

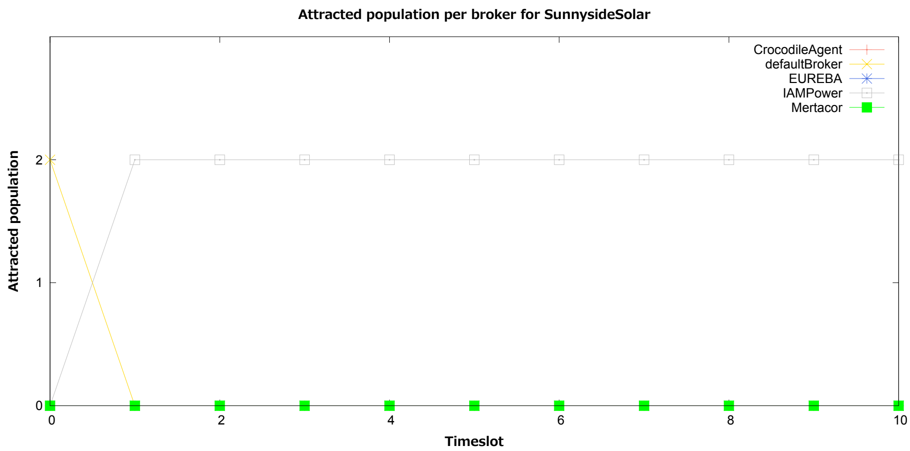

Figure A.10: The distribution of the population of the *SunnysideSolar* customer over the brokers of the pilot competition. This customer has a population of 2.

Appendix B

Pilot customer consumption

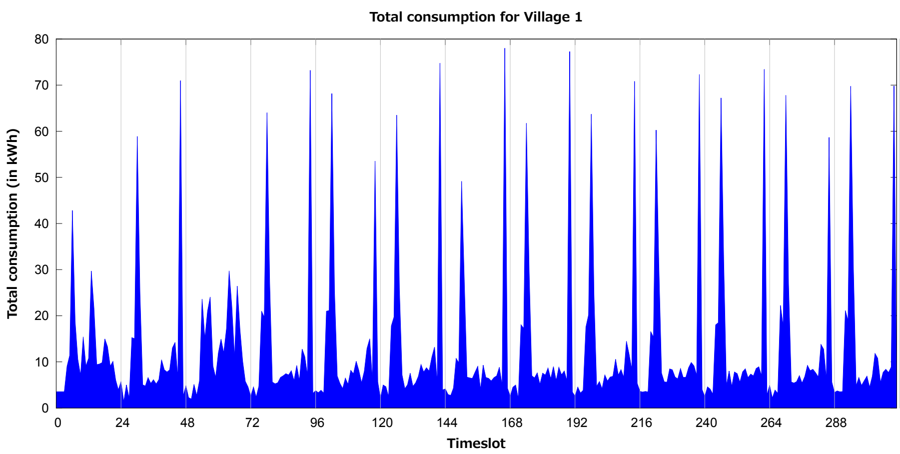

Figure B.1: The total consumption per timeslot for the *Village 1* customer during the pilot competition.

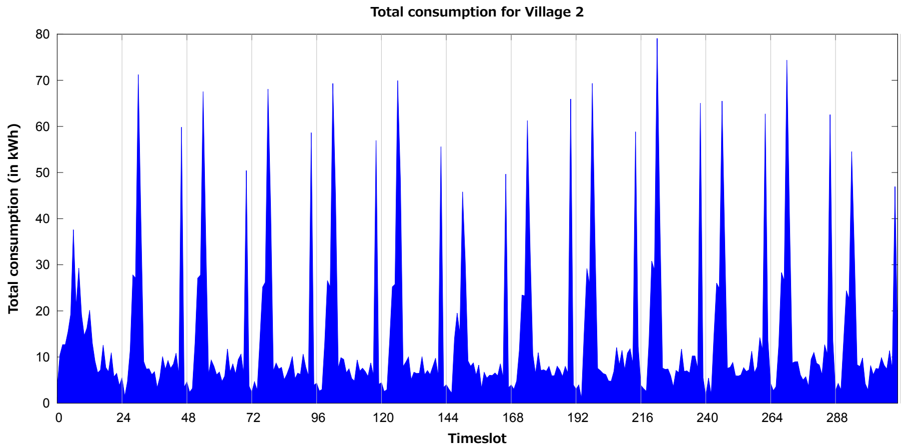

Figure B.2: The total consumption per timeslot for the *Village 2* customer during the pilot competition.

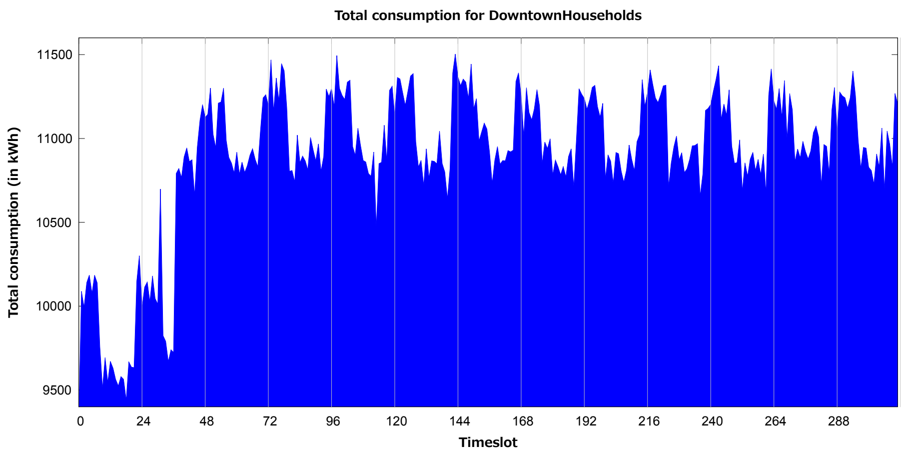

Figure B.3: The total consumption per timeslot for the *DowntownHouseholds* customer during the pilot competition.

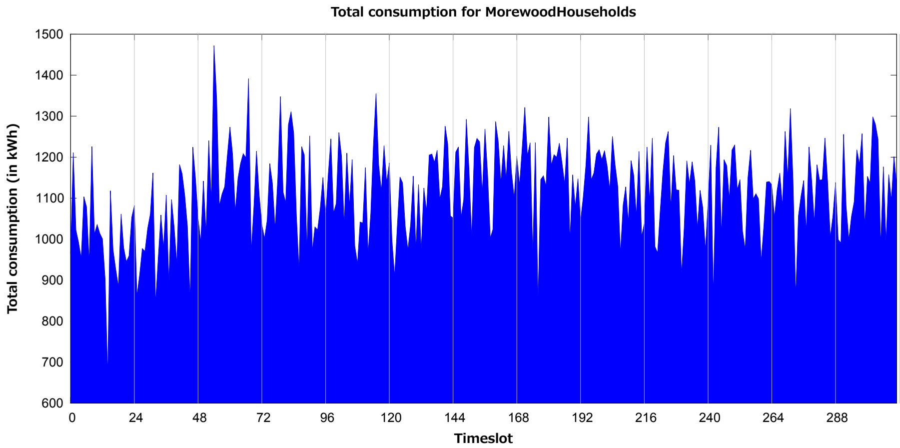

Figure B.4: The total consumption per timeslot for the *MorewoodHouseholds* customer during the pilot competition.

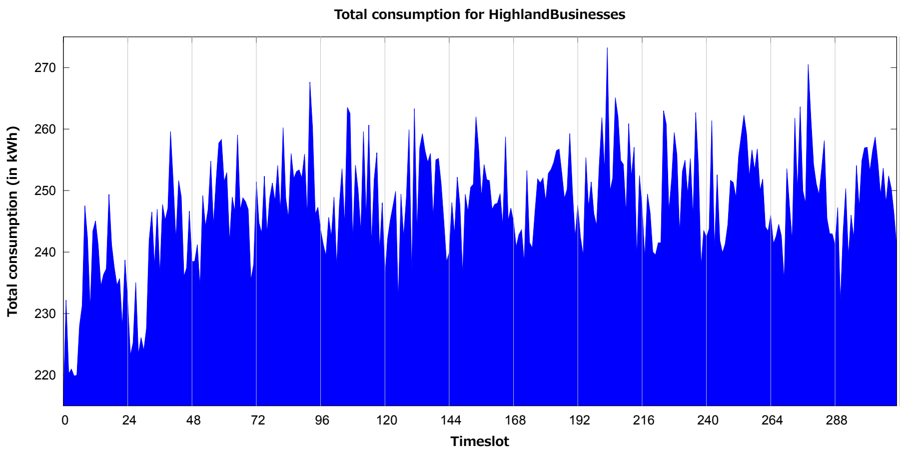

Figure B.5: The total consumption per timeslot for the *HighlandBusinesses* customer during the pilot competition.

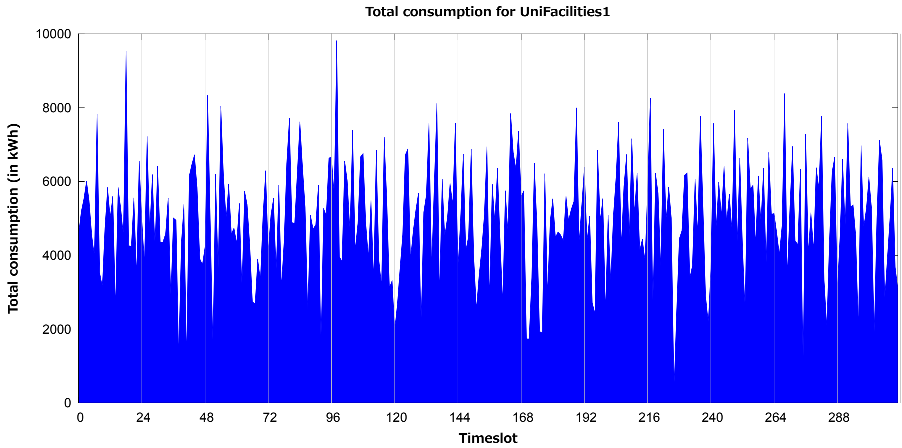

Figure B.6: The total consumption per timeslot for the *UniFacilities1* customer during the pilot competition.

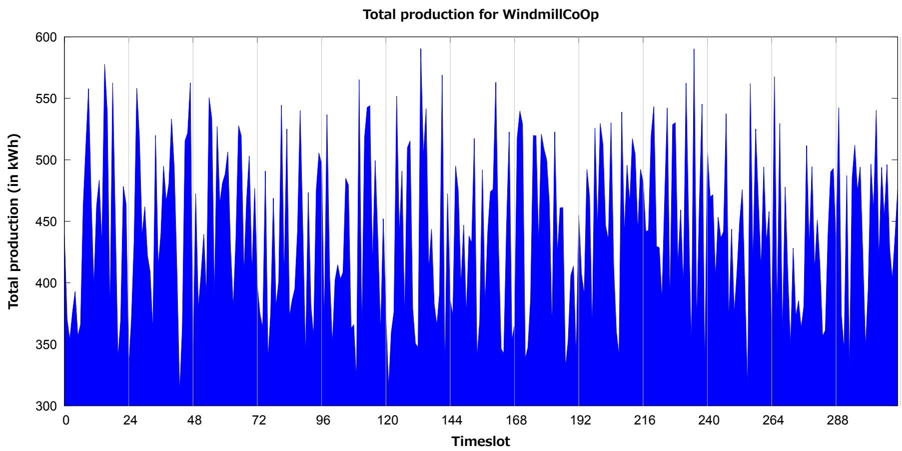

Figure B.7: The total production per timeslot for the *WindmillCoOp* customer during the pilot competition.

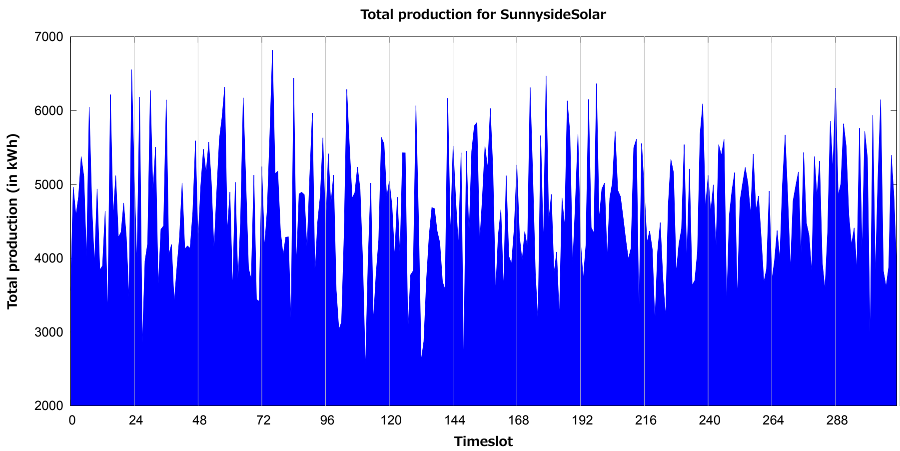

Figure B.8: The total production per timeslot for the *SunnysideSolar* customer during the pilot competition.

Appendix C

Evaluation results

C.1 State-based prediction method

Table C.1: The absolute percentage errors notated as (Q_1APE, MdAPE, Q_3APE) of the predictions for the state-based method given the different scenarios, the number m of timeslots taken into account to form a prediction, and the set of prediction gaps {1, 2, 3, 6, 12, 23}.

Scenario 1.1

m	gap of 1	gap of 2	gap of 3
1	(4.93, 11.41, 22.27)	(5.23, 11.94, 26.22)	(5.49, 12.25, 24.31)
2	(5.12, 11.40, 21.56)	(5.35, **11.91**, 25.75)	(5.27, 12.48, 23.95)
3	(5.11, **11.14**, 21.98)	(5.28, 12.25, 25.89)	(5.22, 12.34, 24.64)
4	(4.75, 11.38, 22.54)	(5.07, 12.30, 25.21)	(5.13, **12.15**, 24.01)
5	(4.77, 11.63, 23.36)	(5.02, 12.07, 25.09)	(5.00, 12.30, 24.20)
6	(4.75, 11.66, 23.10)	(4.93, 12.44, 24.74)	(4.83, 12.21, 24.18)
7	(4.86, 11.62, 23.30)	(4.74, 12.07, 25.00)	(4.12, 12.34, 24.20)

m	gap of 6	gap of 12	gap of 23
1	(5.67, 14.18, 27.50)	(5.60, 13.13, 27.28)	(5.73, 12.89, 24.97)
2	(4.93, 13.57, 24.99)	(5.07, 12.86, 25.02)	(5.25, 11.98, 21.91)
3	(4.81, 13.37, 23.98)	(4.74, 12.66, 23.92)	(5.51, 12.03, 22.81)
4	(4.86, 13.00, 23.95)	(4.81, 12.34, 23.46)	(5.30, **11.88**, 22.61)
5	(4.88, 12.68, 23.79)	(4.80, 12.19, 23.59)	(5.17, 11.94, 22.21)
6	(4.82, 12.46, 23.79)	(5.07, 11.86, 23.62)	(5.36, 12.05, 22.23)
7	(5.03, **12.37**, 23.64)	(5.21, **11.47**, 22.34)	(5.30, 11.89, 22.40)

Scenario 2.1

m	gap of 1	gap of 2	gap of 3
1	(0.63, 1.39, 2.31)	(0.59, 1.40, 2.41)	(0.67, 1.46, 2.50)
2	(0.63, 1.34, 2.18)	(0.65, **1.31**, 2.37)	(0.73, 1.43, 2.45)
3	(0.67, **1.34**, 2.22)	(0.71, 1.31, 2.41)	(0.71, 1.43, 2.43)
4	(0.64, 1.38, 2.24)	(0.67, 1.32, 2.42)	(0.71, 1.38, 2.38)
5	(0.67, 1.40, 2.27)	(0.70, 1.31, 2.42)	(0.68, 1.36, 2.34)
6	(0.68, 1.38, 2.25)	(0.70, 1.33, 2.37)	(0.71, 1.36, 2.37)
7	(0.71, 1.42, 2.22)	(0.72, 1.34, 2.37)	(0.71, **1.35**, 2.34)
m	gap of 6	gap of 12	gap of 23
1	(0.65, 1.41, 2.44)	(0.68, 1.33, 2.44)	(0.78, 1.56, 2.76)
2	(0.64, 1.42, 2.38)	(0.68, 1.35, 2.40)	(0.77, 1.58, 2.70)
3	(0.65, 1.38, 2.37)	(0.68, **1.32**, 2.37)	(0.82, 1.58, 2.72)
4	(0.64, 1.33, 2.40)	(0.71, 1.40, 2.35)	(0.85, 1.53, 2.67)
5	(0.64, 1.32, 2.40)	(0.70, 1.41, 2.35)	(0.83, 1.51, 2.64)
6	(0.65, 1.32, 2.41)	(0.69, 1.40, 2.32)	(0.80, 1.50, 2.60)
7	(0.63, **1.31**, 2.38)	(0.68, 1.43, 2.32)	(0.80, **1.49**, 2.57)

Scenario 2.2

m	gap of 1	gap of 2	gap of 3
1	(0.64, 1.36, 2.37)	(0.62, 1.41, 2.44)	(0.64, 1.57, 2.49)
2	(0.66, 1.33, 2.27)	(0.63, **1.33**, 2.36)	(0.65, 1.47, 2.38)
3	(0.65, 1.35, 2.28)	(0.68, 1.34, 2.35)	(0.63, 1.43, 2.41)
4	(0.63, **1.32**, 2.27)	(0.69, 1.37, 2.32)	(0.63, 1.42, 2.38)
5	(0.65, 1.35, 2.29)	(0.67, 1.37, 2.33)	(0.65, 1.42, 2.30)
6	(0.68, 1.36, 2.28)	(0.69, 1.40, 2.29)	(0.63, 1.42, 2.29)
7	(0.70, 1.36, 2.24)	(0.70, 1.37, 2.25)	(0.63, **1.42**, 2.32)
m	**gap of 6**	**gap of 12**	**gap of 23**
1	(0.68, 1.49, 2.40)	(0.69, 1.45, 2.47)	(0.76, 1.59, 2.67)
2	(0.68, 1.43, 2.32)	(0.68, 1.41, 2.41)	(0.71, 1.55, 2.69)
3	(0.67, 1.42, 2.32)	(0.68, **1.40**, 2.39)	(0.76, 1.52, 2.63)
4	(0.68, 1.41, 2.35)	(0.71, 1.43, 2.35)	(0.74, 1.53, 2.58)
5	(0.69, **1.34**, 2.38)	(0.70, 1.44, 2.36)	(0.71, 1.50, 2.57)
6	(0.68, 1.35, 2.38)	(0.67, 1.46, 2.40)	(0.72, 1.50, 2.55)
7	(0.64, 1.38, 2.37)	(0.66, 1.47, 2.36)	(0.72, **1.46**, 2.52)

Scenario 2.3

m	gap of 1	gap of 2	gap of 3
1	(0.77, 1.46, 2.22)	(0.74, 1.49, 2.40)	(0.71, 1.57, 2.38)
2	(0.72, 1.42, 2.18)	(0.68, 1.39, 2.29)	(0.70, 1.44, 2.33)
3	(0.72, 1.44, 2.13)	(0.70, 1.43, 2.26)	(0.67, 1.40, 2.25)
4	(0.68, **1.41**, 2.03)	(0.71, 1.37, 2.25)	(0.70, 1.34, 2.25)
5	(0.70, 1.44, 2.00)	(0.70, 1.35, 2.26)	(0.72, 1.33, 2.25)
6	(0.75, 1.45, 2.04)	(0.70, **1.34**, 2.25)	(0.74, **1.32**, 2.24)
7	(0.74, 1.44, 2.08)	(0.70, 1.35, 2.25)	(0.75, 1.36, 2.23)

m	gap of 6	gap of 12	gap of 23
1	(0.74, 1.50, 2.43)	(0.70, 1.45, 2.43)	(0.80, 1.49, 2.64)
2	(0.69, 1.50, 2.41)	(0.68, 1.45, 2.37)	(0.76, 1.49, 2.41)
3	(0.66, 1.35, 2.34)	(0.66, **1.40**, 2.29)	(0.71, 1.45, 2.43)
4	(0.71, **1.31**, 2.31)	(0.69, 1.42, 2.26)	(0.71, **1.37**, 2.42)
5	(0.77, 1.34, 2.26)	(0.69, 1.41, 2.26)	(0.68, 1.38, 2.40)
6	(0.78, 1.34, 2.24)	(0.70, 1.42, 2.26)	(0.64, 1.39, 2.39)
7	(0.74, 1.34, 2.23)	(0.73, 1.40, 2.27)	(0.67, 1.39, 2.39)

Scenario 2.4

m	gap of 1	gap of 2	gap of 3
1	(0.65, 1.45, 2.6)	(0.69, 1.49, 2.71)	(0.75, 1.58, 2.79)
2	(0.74, 1.45, 2.59)	(0.74, **1.45**, 2.72)	(0.75, 1.57, 2.79)
3	(0.76, **1.44**, 2.59)	(0.73, 1.48, 2.74)	(0.76, **1.56**, 2.78)
4	(0.72, 1.46, 2.62)	(0.70, 1.53, 2.72)	(0.77, 1.59, 2.79)
5	(0.72, 1.47, 2.65)	(0.71, 1.55, 2.71)	(0.75, 1.58, 2.81)
6	(0.70, 1.51, 2.62)	(0.69, 1.56, 2.70)	(0.72, 1.61, 2.77)
7	(0.70, 1.53, 2.63)	(0.70, 1.60, 2.71)	(0.74, 1.61, 2.79)
m	gap of 6	gap of 12	gap of 23
1	(0.69, 1.65, 2.81)	(0.76, 1.64, 2.99)	(0.79, 1.65, 2.81)
2	(0.70, 1.58, 2.78)	(0.75, 1.65, 2.87)	(0.79, 1.64, 2.73)
3	(0.70, **1.53**, 2.75)	(0.74, 1.63, 2.81)	(0.79, 1.55, 2.71)
4	(0.70, 1.58, 2.71)	(0.76, 1.63, 2.75)	(0.80, 1.52, 2.76)
5	(0.69, 1.59, 2.68)	(0.75, 1.65, 2.66)	(0.78, **1.50**, 2.73)
6	(0.71, 1.56, 2.68)	(0.74, 1.64, 2.67)	(0.76, 1.50, 2.72)
7	(0.73, 1.55, 2.70)	(0.73, **1.62**, 2.65)	(0.76, 1.50, 2.77)

Scenario 3.1

m	gap of 1	gap of 2	gap of 3
1	(1.44, 3.05, 4.83)	(1.46, 3.23, 5.07)	(1.20, 3.24, 5.40)
2	(1.46, 3.02, 4.71)	(1.37, 3.11, 5.04)	(1.38, 3.24, 5.16)
3	(1.40, 3.09, 4.80)	(1.37, 3.11, 5.21)	(1.52, 3.13, 5.18)
4	(1.30, **2.99**, 4.82)	(1.33, 3.13, 5.13)	(1.53, **3.09**, 5.10)
5	(1.32, 3.01, 4.93)	(1.37, 3.12, 5.00)	(1.53, 3.11, 5.21)
6	(1.33, 3.07, 4.90)	(1.32, **3.05**, 5.01)	(1.52, 3.19, 5.21)
7	(1.31, 3.08, 4.90)	(1.40, 3.05, 4.99)	(1.51, 3.29, 5.17)
m	gap of 6	gap of 12	gap of 23
1	(1.30, 3.44, 5.42)	(1.85, 3.38, 5.45)	(1.55, 3.35, 5.41)
2	(1.33, 3.36, 5.22)	(1.71, 3.44, 5.48)	(1.56, 3.24, 5.24)
3	(1.39, 3.39, 5.31)	(1.69, **3.37**, 5.54)	(1.63, 3.24, 5.21)
4	(1.42, **3.32**, 5.34)	(1.60, 3.41, 5.35)	(1.46, 3.19, 5.28)
5	(1.44, 3.34, 5.40)	(1.57, 3.47, 5.39)	(1.63, **3.14**, 5.32)
6	(1.47, 3.38, 5.40)	(1.49, 3.45, 5.39)	(1.55, 3.21, 5.27)
7	(1.47, 3.40, 5.45)	(1.47, 3.37, 5.37)	(1.58, 3.29, 5.26)

Scenario 4.1

m	gap of 1	gap of 2	gap of 3
1	(8.15, 16.27, 25.80)	(8.80, 16.98, 27.96)	(8.04, 17.58, 28.47)
2	(8.32, 15.79, 25.99)	(8.38, **15.54**, 28.68)	(8.53, 16.72, 28.38)
3	(8.53, 15.81, 26.23)	(8.39, 15.91, 27.88)	(8.54, 16.68, 28.31)
4	(7.88, 16.03, 26.78)	(8.02, 16.60, 26.40)	(8.85, 16.69, 28.38)
5	(8.16, 16.13, 26.10)	(8.57, 16.48, 27.08)	(8.85, **16.35**, 28.53)
6	(8.30, 15.92, 25.59)	(8.72, 16.10, 27.80)	(8.80, 16.94, 28.50)
7	(8.41, **15.71**, 25.69)	(8.59, 16.67, 27.63)	(8.85, 17.43, 28.33)

m	gap of 6	gap of 12	gap of 23
1	(9.59, 18.49, 30.06)	(8.89, **17.03**, 31.25)	(8.99, 17.92, 30.68)
2	(8.97, 18.28, 28.91)	(9.32, 17.38, 28.68)	(8.66, 17.46, 29.48)
3	(8.77, 18.18, 29.79)	(9.48, 17.30, 29.92)	(8.43, **17.29**, 29.16)
4	(9.46, 18.18, 29.73)	(9.49, 17.65, 29.85)	(8.38, 17.84, 28.25)
5	(9.67, 17.92, 28.73)	(9.34, 17.79, 29.23)	(8.71, 17.61, 27.85)
6	(9.51, 17.74, 28.66)	(9.32, 17.77, 29.40)	(8.99, 17.62, 27.45)
7	(9.37, **17.67**, 28.74)	(9.29, 17.61, 28.30)	(9.39, 17.48, 26.78)

Scenario 5.1

m	gap of 1	gap of 2	gap of 3
1	(0.84, 1.84, 3.28)	(0.92, 2.02, 3.57)	(0.90, 1.88, 3.56)
2	(0.93, 1.85, 3.20)	(0.89, 1.95, 3.34)	(0.99, 1.90, 3.47)
3	(0.88, **1.80**, 3.12)	(0.91, 1.92, 3.37)	(0.95, **1.83**, 3.33)
4	(0.81, 1.88, 3.09)	(0.90, 1.84, 3.32)	(0.92, 1.91, 3.30)
5	(0.82, 1.88, 3.10)	(0.90, **1.83**, 3.35)	(0.92, 1.89, 3.29)
6	(0.85, 1.93, 3.13)	(0.92, 1.85, 3.25)	(0.94, 1.86, 3.24)
7	(0.88, 1.89, 3.09)	(0.88, 1.89, 3.24)	(0.97, 1.87, 3.21)

m	gap of 6	gap of 12	gap of 23
1	(0.90, 2.02, 3.39)	(0.97, 2.06, 3.94)	(1.07, 2.15, 4.42)
2	(0.92, 1.89, 3.21)	(1.01, 1.93, 3.71)	(1.04, 1.96, 3.72)
3	(0.91, 1.90, 3.35)	(0.96, 1.99, 3.60)	(0.97, 1.88, 3.55)
4	(0.88, 1.89, 3.33)	(0.98, 1.93, 3.53)	(0.93, 1.87, 3.45)
5	(0.91, 1.89, 3.31)	(0.92, 1.87, 3.50)	(0.91, 1.85, 3.37)
6	(0.93, 1.89, 3.33)	(0.91, 1.86, 3.40)	(0.91, 1.87, 3.35)
7	(0.95, **1.87**, 3.34)	(0.92, **1.85**, 3.39)	(0.88, **1.82**, 3.36)

C.2 Regression tree method

Table C.2: The average absolute percentage errors notated as (Q_1APE, MdAPE, Q_3APE) of the predictions for the regression tree prediction method given the different scenarios and the set of prediction gaps {1, 2, 3, 6, 12, 23}, based on ten different runs per scenario. The lower and upper bound of each set of ten predictions associated to one scenario are also displayed.

Scenario 1.1

Run	gap of 1	gap of 2	gap of 3
L.bound	(5.91, 13.36, 23.61)	(5.75, 13.23, 23.67)	(5.84, 13.20, 23.46)
Avg	(6.32, 13.65, 24.09)	(6.16, 13.45, 24.18)	(6.21, 13.52, 24.00)
U.bound	(6.73, 13.94, 24.56)	(6.56, 13.67, 24.69)	(6.59, 13.84, 24.54)
Run	gap of 6	gap of 12	gap of 23
L.bound	(5.88, 13.29, 23.74)	(6.07, 13.46, 24.16)	(6.64, 13.87, 23.98)
Avg	(6.28, 13.63, 24.42)	(6.53, 13.91, 24.61)	(6.85, 14.16, 24.66)
U.bound	(6.68, 13.97, 25.10)	(7.00, 14.35, 25.05)	(7.06, 14.46, 25.34)

Scenario 2.1

	gap of 1	gap of 2	gap of 3
L.bound	(0.68, 1.50, 2.44)	(0.67, 1.48, 2.46)	(0.68, 1.46, 2.45)
Avg	(0.71, 1.52, 2.48)	(0.71, 1.51, 2.50)	(0.71, 1.49, 2.48)
U.bound	(0.74, 1.55, 2.53)	(0.74, 1.53, 2.54)	(0.74, 1.53, 2.52)
	gap of 6	gap of 12	gap of 23
L.bound	(0.70, 1.47, 2.48)	(0.66, 1.47, 2.45)	(0.72, 1.49, 2.51)
Avg	(0.74, 1.49, 2.52)	(0.69, 1.49, 2.49)	(0.75, 1.53, 2.54)
U.bound	(0.77, 1.52, 2.55)	(0.72, 1.51, 2.52)	(0.78, 1.58, 2.57)

Scenario 2.2

	gap of 1	gap of 2	gap of 3
L.bound	(0.67, 1.45, 2.46)	(0.70, 1.50, 2.48)	(0.69, 1.48, 2.47)
Avg	(0.72, 1.50, 2.53)	(0.74, 1.53, 2.53)	(0.73, 1.53, 2.53)
U.bound	(0.76, 1.55, 2.60)	(0.77, 1.57, 2.57)	(0.77, 1.58, 2.58)
	gap of 6	gap of 12	gap of 23
L.bound	(0.69, 1.52, 2.50)	(0.67, 1.48, 2.46)	(0.74, 1.51, 2.50)
Avg	(0.74, 1.56, 2.56)	(0.70, 1.51, 2.52)	(0.76, 1.55, 2.54)
U.bound	(0.80, 1.60, 2.61)	(0.74, 1.54, 2.58)	(0.79, 1.59, 2.58)

Scenario 2.3

	gap of 1	gap of 2	gap of 3
L.bound	(0.70, 1.40, 2.30)	(0.71, 1.39, 2.29)	(0.71, 1.39, 2.29)
Avg	(0.73, 1.44, 2.37)	(0.73, 1.42, 2.35)	(0.74, 1.43, 2.34)
U.bound	(0.76, 1.47, 2.43)	(0.75, 1.46, 2.40)	(0.77, 1.48, 2.39)
	gap of 6	gap of 12	gap of 23
L.bound	(0.72, 1.41, 2.32)	(0.71, 1.46, 2.25)	(0.68, 1.50, 2.34)
Avg	(0.76, 1.45, 2.37)	(0.74, 1.52, 2.32)	(0.72, 1.53, 2.39)
U.bound	(0.79, 1.49, 2.41)	(0.77, 1.58, 2.39)	(0.76, 1.56, 2.45)

Scenario 2.4

	gap of 1	gap of 2	gap of 3
L.bound	(0.69, 1.46, 2.51)	(0.67, 1.45, 2.47)	(0.67, 1.45, 2.47)
Avg	(0.73, 1.50, 2.55)	(0.71, 1.51, 2.52)	(0.71, 1.49, 2.51)
U.bound	(0.77, 1.55, 2.59)	(0.75, 1.56, 2.57)	(0.75, 1.53, 2.55)
	gap of 6	gap of 12	gap of 23
L.bound	(0.67, 1.45, 2.42)	(0.69, 1.48, 2.43)	(0.70, 1.57, 2.59)
Avg	(0.69, 1.49, 2.45)	(0.72, 1.51, 2.47)	(0.74, 1.61, 2.64)
U.bound	(0.72, 1.52, 2.48)	(0.74, 1.54, 2.51)	(0.77, 1.65, 2.70)

Scenario 3.1

	gap of 1	gap of 2	gap of 3
L.bound	(1.46, 3.16, 5.15)	(1.42, 3.15, 5.20)	(1.42, 3.14, 5.19)
Avg	(1.52, 3.23, 5.26)	(1.48, 3.22, 5.33)	(1.50, 3.22, 5.31)
U.bound	(1.57, 3.30, 5.36)	(1.55, 3.28, 5.45)	(1.58, 3.30, 5.43)
	gap of 6	gap of 12	gap of 23
L.bound	(1.42, 3.17, 5.20)	(1.36, 3.17, 5.32)	(1.41, 3.12, 5.18)
Avg	(1.49, 3.25, 5.31)	(1.43, 3.24, 5.43)	(1.50, 3.20, 5.25)
U.bound	(1.57, 3.32, 5.42)	(1.50, 3.31, 5.54)	(1.58, 3.28, 5.32)

Scenario 4.1

	gap of 1	gap of 2	gap of 3
L.bound	(8.31, 16.15, 26.87)	(8.56, 16.28, 26.95)	(8.28, 16.19, 26.93)
Avg	(8.62, 16.59, 27.41)	(8.80, 16.72, 27.62)	(8.64, 16.50, 27.56)
U.bound	(8.92, 17.03, 27.96)	(9.03, 17.16, 28.28)	(9.00, 16.82, 28.19)
	gap of 6	gap of 12	gap of 23
L.bound	(8.18, 16.32, 26.90)	(8.37, 16.08, 27.57)	(8.01, 15.76, 26.27)
Avg	(8.55, 16.58, 27.49)	(8.74, 16.54, 27.99)	(8.37, 16.19, 26.74)
U.bound	(8.92, 16.84, 28.08)	(9.10, 17.00, 28.40)	(8.73, 16.62, 27.21)

Scenario 5.1

	gap of 1	gap of 2	gap of 3
L.bound	(1.07, 2.29, 4.12)	(1.09, 2.28, 4.04)	(1.06, 2.25, 3.93)
Avg	(1.13, 2.34, 4.22)	(1.14, 2.33, 4.19)	(1.13, 2.31, 4.09)
U.bound	(1.20, 2.39, 4.31)	(1.20, 2.38, 4.33)	(1.19, 2.37, 4.25)
	gap of 6	gap of 12	gap of 23
L.bound	(1.04, 2.13, 3.73)	(1.02, 2.12, 3.78)	(0.95, 2.14, 3.89)
Avg	(1.10, 2.20, 3.93)	(1.07, 2.20, 3.89)	(1.00, 2.22, 4.03)
U.bound	(1.16, 2.27, 4.12)	(1.12, 2.28, 3.99)	(1.06, 2.29, 4.16)

Bibliography

[1] Madison Gas and Electric - Facility Efficiency Advice. Technical report, Madison Gas and Electric, Dec. 2011. `http://www.mge.com/business/saving/BEA/_escrc_0013000000DP22YAAT-2_BEA1_CEA.html`.

[2] R. E. Abdel-Aal. Univariate Modeling and Forecasting of Monthly Energy Demand Time Series Using Abductive and Neural Networks. *Comput. Ind. Eng.*, 54(4):903–917, May 2008.

[3] D. G. Altman. *Practical Statistics for Medical Research*. Chapman & Hall, 1991.

[4] J. S. Armstrong and F. Collopy. Error Measures for Generalizing About Forecasting Methods: Empirical Comparisons. *International Journal of Forecasting*, 8(1):69–80, June 1992.

[5] L. Breiman. Random Forests. In *Machine Learning*, pages 5–32, 2001.

[6] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984.

[7] J. L. Brenner, D. A. D'Esopo, and A. G. Fowler. Difference Equations in Forecasting Formulas. *Management Science*, 15(3):141–159, 1968.

[8] S. Franklin and A. Graesser. Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents. In *Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages*, pages 21–35. Springer-Verlag, 1996.

[9] E. S. Gardner. Exponential Smoothing: The State of the Art. *Journal of Forecasting*, 4(1):1–28, 1985.

[10] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.

[11] P. Goodwin and R. Lawton. On the Asymmetry of the Symmetric MAPE. *International Journal of Forecasting*, 15(4):405–408, Oct. 1999.

[12] K. Green and L. Tashman. Percentage Error: What Denominator? *Foresight: The International Journal of Applied Forecasting*, (12):36–40, 2009.

[13] A. Greenwald. The 2002 Trading Agent Competition: An Overview of Agent Strategies. *AI Magazine*, 24:83–91, 2002.

[14] J. Hoover. Measuring Forecast Accuracy: Omissions in Today's Forecasting Engines and Demand-Planning Software. *Foresight: The International Journal of Applied Forecasting*, (4):32–35, 2006.

[15] R. J. Hyndman. Another Look at Forecast-Accuracy Metrics for Intermittent Demand. *Foresight: The International Journal of Applied Forecasting*, (4):43–46, 2006.

[16] R. J. Hyndman and A. B. Koehler. Another Look at Measures of Forecast Accuracy. *International Journal of Forecasting*, pages 679–688, 2006.

[17] G. V. Kass. An Exploratory Technique for Investigating Large Quantities of Categorical Data. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 29(2):119–127, 1980.

[18] W. Ketter, J. Collins, and C. A. Block. Smart Grid Economics: Policy Guidance through Competitive Simulation. Research Paper ERS-2010-043-LIS, Erasmus Research Institute of Management (ERIM), 2010.

[19] W. Ketter, J. Collins, P. P. Reddy, and C. M. Flath. The Power Trading Agent Competition. Research Paper ERS-2011-011-LIS, Erasmus Research Institute of Management (ERIM), 2011.

[20] H. Küchenhoff and R. J. Carroll. Segmented regression with erros in predictors: Semi-parametric and parametric methods. *Statistics in Medicine*, 16(2):169–188, 1997.

[21] S. Makridakis, A. Anderson, R. Carbone, R. Fildes, M. Hibon, Lewandowskiand, E. Parzen, and R. Winkler. The Accuracy of Extrapolation (Time Series) Methods: Results of a Forecasting Competition. *Journal of Forecasting*, 1:111–153, 1982.

[22] S. Makridakis and M. Hibon. The M3-Competition: Results, Conclusions and Implications. *International Journal of Forecasting*, 16(4):451–476, 2000.

[23] W. S. McCulloch and W. Pitts. A Logical Calculus of the Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics*, 5:115–133, 1943.

[24] R. H. Myers. *Classical and Modern Regression with Applications.* PWS-KENT Publishing Company, Boston, MA, second edition, 1994.

[25] C. C. Pegels. Exponential Forecasting: Some New Variations. *Management Science*, 15:311–315, 1969.

[26] J. V. Ringwood, D. Bofelli, and F. T. Murray. Forecasting Electricity Demand on Short, Medium and Long Time Scales Using Neural Networks. *J. Intell. Robotics Syst.*, 31(1-3):129–147, May 2001.

[27] S. A. Roberts. A General Class of Holt-Winters Type Forecasting Models. *Management Science*, 28(7):808–820, 1982.

[28] D. Rumelhart, G. Hintont, and R. Williams. Learning Representations by Back-Propagating Errors. *Nature*, 323(6088):533–536, 1986.

[29] G. Shmueli, N. Patel, and P. Bruce. *Data Mining for Business Intelligence: Concepts, Techniques, And Applications in Microsoft Office Excel With XLMiner*. John Wiley & Sons, 2006.

[30] J. W. Taylor. Exponential Smoothing with a Damped Multiplicative Trend. *International Journal of Forecasting*, 19(4):715–725, 2003.

[31] P. R. Winters. Forecasting Sales by Exponentially Weighted Moving Averages. *Management Science*, 6(3):324–342, April 1960.

[32] L. A. Zadeh. Fuzzy Logic. *Computer*, 21:83–93, 1988.