Managing Power Conservation in Wireless Networks

Kongluan Lin, John Debenham[1], and Simeon Simoff[2]

[1] QCIS, FEIT, University of Technology, Sydney, Australia
[2] Computing and Mathematics, UWS, Sydney, Australia

Abstract. A major project is investigating methods for conserving power in wireless networks. A component of this project addresses methods for predicting whether the user demand load in each zone of a network is increasing, decreasing or approximately constant. These predictions are then fed into the power regulation system. This paper describes a real-time predictive model of network traffic load which is derived from experiments on real data. This model combines a linear regression based model and a highly reactive model that are applied to real-time data that is aggregated at two levels of granularity. The model gives excellent performance predictions when applied to network traffic load data.

Key words: power conservation, **data mining**

1 Introduction

Power conservation in wireless networks is attracting considerable attention. Most approaches are based either on maximising the number of stations that can be put into sleep mode, or on minimising the power for transmission when the unit is active [1]. However, the majority of approaches are not based on predictive models of network load. This paper describes work in a power conservation project conducted in collaboration between UTS and Alcatel-Lucent (Bell Labs). A component of this project addresses methods for predicting user demand load in each zone of a network. This paper describes the data mining methods used to build a predicative model that is then fed into the power regulation system.

The size and nature of communications networks suggests that forecasting methods should be distributed and should react quickly. We use the term *agent* to refer to a decision making entity located in each zone of the network. Each agent has to determine: which signals to select, and how the selected signals should be combined. Data mining can be an expensive business; costly solutions may be justified if they can be replicated across the network. The management of the extent to which data mining is applied is important, and is not discussed directly here.

The model described aims only to predict whether the local load is *either* generally unchanged, *or* is increasing, *or* is decreasing. This model combines:

- signals derived from local observations that each agent makes *within* its zone in the network,
- signals from agents in neighbouring zones, and
- signals derived from background information sources *external to* the network

The method for deriving predictions from local observations is particularly interesting. Here we found that a subtle combination of a linear regression based model and a highly reactive model applied to real-time data that is aggregated at two levels of granularity outperformed non-linear methods. This technique is described in detail.

Section 2 discusses the application of data mining to network management as well as distributed data mining techniques that are particularly relevant to this work. Section 3 explains how three types of signal are combined to form our solution. Section 4 describes the experimental case study on real-time mining of local observations using real network data. The data was selected from Internet traffic at the University of Technology, Sydney, where we found that data from some of the University's zones had characteristics similar to wireless networks in that it was bursty with fluctuations that were significant but bounded. Finally, Section 5 concludes.

2 Background

There is an established history of data mining in network management. We discuss work on alarm or fault related issues as well as work in distributed data mining.

2.1 Mining Alarms and Faults in Communications Networks

Alarm Correlation Alarms are messages produced by different components of networks. They describe some sort of abnormal situations [2]. Modern communication networks produce large numbers of alarm messages. These alarm messages traversing the network burden the network traffic, possibly lead to packet loss, latency and data retransmission, and ultimately degrade the network performance [3]. Also, due to the rapid development of hardware and software used in communication networks, the characteristics of the alarm sequences are changed as new nodes are added to the network or old ones are updated [4]. Thus, the operators may not have time to learn how to respond to each situation appropriately. In order to avoid overloading operators, alarm correlation systems are used to filter and condense the incoming alarms and diagnose the initial cause of the alarm burst [5]. Due to the dynamic nature of growing telecommunication networks, alarm correlation systems need to adapt to different topologies and extensions of network structure [5]. Also, they need to be tolerant of missing and unnecessary alarm data which are defined as noise.

Neural Networks have characteristics which makes them suitable for the alarm correlation task. No expert knowledge is needed to train the neural network. Moreover, Neural Networks are resistant to noise because of their generalising capabilities [5]. The "Cascade Correlation Alarm Correlator" (CCAC) is a neural network based alarm correlator. It minimises the count of operations — no topology of the hidden layer has to be proposed during training. The system is able to treat noise with up to 25 percent of missing alarms while still achieving 99.76% correct decisions [5]. Self-Organising Maps, a type of neural network, are able to recognise input alarm patterns even when the input data is noisy, corrupted or has significant variation [6].

Bayesian Belief Networks (BBN) are well suited to automated diagnosis because of their deep representation of knowledge and their precise calculations [7]. A BBN represents cause and effect between observable symptoms and the unobserved problems.

When a set of symptoms are observed, the most likely cause can be determined. However, the development of a diagnostic BBN requires a deep understanding of the cause and effect relationships in a domain, provided by domain experts [7].

Fault prediction The occurrence of a fault often triggers alarm signals. When two consecutive faults occur within a short time, the alarms corresponding to them may mix together. Fault identification may be a very difficult task when the operator is required to take into account the network elements up or down stream of the fault that are also issuing alarms.

The "fixed time windows" method has been used by Sasisekharan and others to predict network faults [3]. The basic idea is that a consecutive period of time is divided into two windows W_a and W_b first, then the measurements made in W_a is used to predict problems in W_b [3]. Because faults are often transient, a reasonably long period for W_b should be specified.

The "Telecommunication Alarm Sequence Analyser" (TASA) has also been used for fault prediction. By using a specialised data mining algorithm, this tool can discover recurrent patterns of alarms automatically [8]. Network specialists then use this information to construct a rule-based alarm correlation system, which can then be used to identify faults in real time. Also, TASA is capable of finding episodic rules that depend on temporal relationships between the alarms [8]. For example, it may discover the following rule: if alarms of type "link alarm" and type "link failure" occur within 5 seconds, then an alarm of type "high fault rate" occurs within 60 seconds with a probability of 0.7 [9]. Based on the rules, faults can be predicted, and counter-measures can be taken to deal with the faults in advance.

"Timeweaver" is another tool for fault prediction. It is a genetic algorithm based machine learning system that predicts rare events by identifying predictive temporal and sequential patterns [10]. It consists of two processes. First, a Genetic Algorithm is used to extract alarm patterns and then a greedy algorithm is applied to form prediction rules. Compared with some existing methods like ANSWER, RIPPER, C4.5, Timeweaver performs better at the prediction task [10].

2.2 Distributed Data Mining of Communications Networks

Traditionally data mining was based on a centralised approach. However, in distributed computing environments, such as the Internet, intranets, sensor networks, wireless networks and Peer-to-peer systems, it is often desirable to mine data that is distributed in different places [11]. In such cases, centralised data mining approach is inappropriate because of its long response time and inability to capitalise on distributed resources [11]. Distributed data mining is often indicated in distributed environments. For example, in a wireless sensor network with limited communication bandwidth and limited battery power, the central collection of data from every sensor node may create heavy traffic and consume a considerable amount of power. In contrast, distributed data mining may be more suitable because it reduces the communication load and spreads power consumption evenly across the different nodes of the sensor network.

Distributed data mining can also be used in network management, and for bundled service management. It offers the following advantages over centralised mining:

Table 1. Data description

Atribute	Description
Date Time	Date time
Traffic in (Volume)	Sum of inbound network load
Traffic in (Volume) (Raw)	Sum of inbound network load
Traffic out (volume)	Sum of outbound network load
Traffic out (volume) (Raw)	Sum of outbound network load

1. Network traffic and processing load in the network management system may be both reduced by performing data processing closer to the network elements.
2. Distributed data mining methods may scale well were centralised methods will not.
3. Searches can be performed closer to the data, improving speed and efficiency.
4. Distributed network management may be inherently more robust without depending on continuous communications between the network management system and network elements [12].

Recently, agent-based distributed data mining has become a very active research area — this combines distributed mining with distributed decision making. Ogston and Vassiliadis use agents to simulate a peer-to-peer auction and a centralised auction [13] for resource allocation. They show that the distributed auction exhibits price convergence behaviour similar to that of the centralised auction. Also, the peer-to-peer system has a constant cost in the number of message rounds needed to find the market equilibrium price as the number of traders is increased, in contrast to the linear cost incurred by the central auctioneer [13]. In terms of message costs, the peer-to-peer system outperforms the central auction.

BODHI, implemented in Java, has been designed for collective DM tasks on heterogeneous data sites [14]. This framework requires low network communication within local and global data models. The mining process is distributed to the local agents and mobile agents that carry data and knowledge. A central facilitator agent is responsible for initialising and coordinating the data mining task within the agents.

Parallel Data Mining Agents (PADMA) architecture is proposed by Kargupta, Hamzaoglu and Stafford [15]. PADMA deals with the problem of distributed data mining from homogeneous data sites. At first, data cluster models are counted by agents locally at different sites. Then, the local models are collected to a central site to perform a second-level clustering to produce the global cluster model.

Papyrus is a Java-based system addressing wide-area distributed data mining over clusters of heterogeneous data sites and meta-clusters [16]. It supports predictive model strategies including C4.5. Mobile data mining agents move data, intermediate results, and models between clusters to perform all computation locally or from local sites to a central root which produces the final result [16]. Each cluster has one distinguished node which acts as its cluster access and control point for the agents. Coordination of the overall clustering task is either done by a central root site or the (peer-to-peer) network of cluster access points.

All the above approaches aim to integrate the knowledge which is extracted from data at different geographically distributed network sites with a minimum amount of

Table 2. Correlation coefficient 1-minute time granularity

Coefficient between:	In(T)/In(T-1)	In(T)/In(T-1)	In(T)/In(T-2)
Granularity:	1 minute	5 minutes	5 minutes
B1	0.89	0.67	0.50
B3	0.88	0.66	0.48
B4	0.83	0.76	0.61
B5	0.90	0.82	0.69
B10	0.95	0.79	0.58

network communication, and maximum of local computation. Concerning distributed data mining algorithms, Bandyopadhyay and others [17] have introduced a P2P K-Means algorithm for distributed clustering of data streams in a peer-to-peer sensor network environment. In the P2P K-Means algorithm, computation is performed locally, and communication of the local data models is restricted only within a limited neighbourhood. As opposed to the full synchronisation required in some other algorithms, synchronisation in P2P K-Means is restricted only within a neighbourhood. Moreover, even if some node and/or link fails, the algorithm can continue, though its performance will degrade gracefully with an increase in the number of failures.

Although distributed data mining approach could be very useful in network management, associated privacy issues are important. Roughan and Zhang proposed a distributed data mining algorithm to conduct summarisation of Internet traffic without revealing traffic volume of any ISP [18].

3 Solution Structure

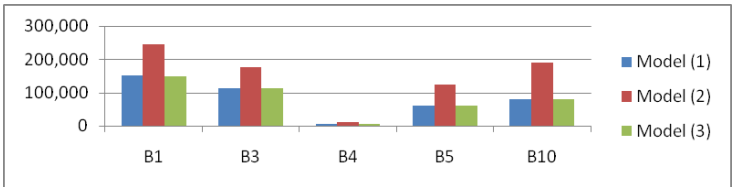
Any intelligent approach to conserve power will incorporate predictive models of load. We combine three solutions to smaller problems to form the complete solution:

1. Real-time mining of local observations that predicts the load in a zone on the basis of observed variations in load on the stations in that zone.
2. Off-line data mining is applied to historic load data over a region to identify whether changes in load in one zone may, under certain conditions, signal subsequent changes in another zone. For example, the natural flow of pedestrian traffic around a building may be the underlying cause of such a relationship.
3. Background mining of data, text and news sources that are *outside* the network. For example, knowledge that a football match may be held in a certain stadium next Saturday may have significant implications for load near the stadium.

The important point is that any solution to the above problem should attempt to capitalise on these three approaches. Also, the solutions must be scalable and operate fast. Scalability suggests distributed decision making systems, and speed of operation suggests they should be based on simple models with low computational demands.

Our solution combines three classes of signals: first, signals that an agent derives from observations within its zone; second, signals that an agent chooses to import from adjacent zones, and third, signals that an agent chooses to import from the mining of

Fig. 1. Goodness of fit generated by Model (1), (2) and (3)



background data. Traditional data mining techniques are applied to select and combine these three classes of signal — as we noted in Section 1 this is a costly process unless the solution derived can be replicated across the network.

4 Real-time mining of local observations

Our overall aim is to identify a suitable architecture for load prediction in LTE networks. To do this, we based our experiments on Ethernet load data obtained from University of Technology, Sydney (UTS), because the data is characteristic of load data in LTE networks. Our hypothesis is that the solution to the prediction of UTS Ethernet load will indicate the architecture for load prediction in LTE networks.

In each zone of UTS, an agent is used to record the network load in the zone over a period of time, and then analyses it to build a load predictive model for the zone. Based on the predictive model, each agent can then forecast its zone's network load. In this experiment, only predictive models for inbound network load are introduced, since outbound network load can be predicted similarly. Also, predictive models are built by applying linear regression and moving average[3] method is used to smooth coarse data to improve prediction accuracy. Besides, we use the goodness of fit as a measure of the accuracy of predictive models, which is defined as follows:

$$\sigma_{est} = \sqrt{\frac{\sum (Y - Y')^2}{N}}$$

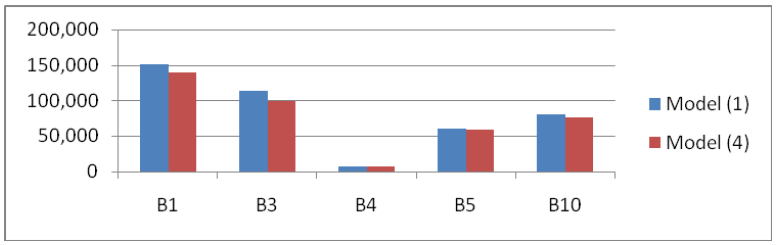
Where *Y* is an actual value and *Y'* is a predicted value while *N* is the number of values to predict.

4.1 Data pre-processing and Transformation

A key feature of the method described is that it operates on data aggregated at two levels of granularity. In the Internet load data used in the experiments the aggregation periods were 1 minute and 5 minutes. These granularity settings were derived as a result of visual examination of the raw data. If the method described is applied to wireless

[3] Subset size for moving average method is arbitrarily assigned to 5, and moving average for inbound network load is calculated as: $A_{\text{in}}(T) = \frac{\text{In}(T-1)+\text{In}(T-2)+\text{In}(T-3)+\text{In}(T-4)+\text{In}(T-5)}{5}$

Fig. 2. Comparison of Goodness of fit between Model (1) and (4)



network traffic then these levels would of course be considerably shorter, and would be derived by visual examination of data as was performed in these experiments.

Table 1 shows the column headings for the raw data — some columns have the same meaning. For example, Traffic in (Volume) and Traffic in (Volume) (Raw) represent the same information, both of raws stand for the inbound network traffic volume within a minute but measured by different units, Kilobyte and Byte respectively. Since they represent the same information, only one of the two columns is considered. In our experiments, data is reconstructed with only three columns: Data Time, Traffic In, and Traffic Out.

The University of Technology, Sydney has Internet facilities deployed through a large campus. The university monitors data in a number of *zones* that vary from teaching laboratories to open access areas in the various building foyers. We inspected the data and selected six zones on the basis that they exhibited characteristics of wireless data in that it was bursty with fluctuations that were significant buy bounded. These zones are referred to in this paper as B1, B3, B4, B5 and B10 — these labels have no significance beyond identifying the data sets selected.

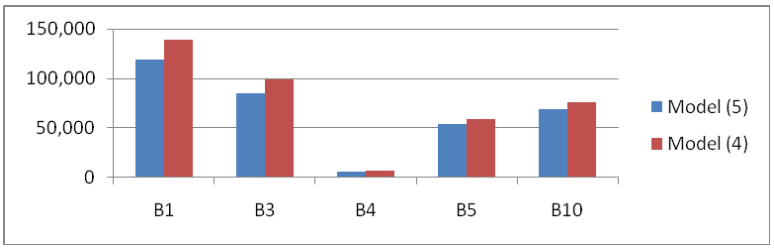
Initial analysis of the data applied Pearson's *sample coefficient of linear correlation* — a measure of the tendency of two variables to vary together — was applied to the five data sets. It is often denoted by r and defined as follows:

$$r = \frac{n\sum_i X_i Y_i - \sum_i X_i \times \sum_i Y_i}{\sqrt{n\sum_i X_i^2 - (\sum_i X_i)^2} \times \sqrt{n\sum_i Y_i^2 - (\sum_i Y_i)^2}}$$

An r value of 1 or -1 indicates a perfect linear relationship between variable X and Y, while value of 0 means variable X is independent of Y. If $r = 1$ then Y always increases as X increases. If $r = -1$ then Y always decreases as X increases.

The results of this initial analysis are shown in Table 2. For 1 minute granularity the correlation coefficient between In(T) and In(T-1) is very high, ranging from 0.83 to 0.95 within the five zones. The correlation coefficient between In(T) and In(T-1) based on 1 minute granularity data is greater than that based on 5 minute granularity data. Consequently we hypothesise that current network traffic load has a strong linear relationship with preceding network traffic load. That is to say, the network load recorded in the previous minute can be used to predict the network load in the next minute. Also,

Fig. 3. Comparison of Goodness of fit between Model (4) and (5)



data of 1 minute granularity is a better foundation for predictive models than data of 5 minute granularity — this is consistent with intuition.

4.2 Models Based on Moving Averages

We denote the total inbound and outbound network load within the minute T as $\mathrm{In}(T)$ and $\mathrm{Out}(T)$ respectively. An intuitive assumption is that $\mathrm{In}(T)$ could have strong linear relationships with its previous moving average like $A_{\mathrm{in}}(T-1)$ and $A_{\mathrm{in}}(T-2)$, and it may also have linear relationship with $A_{\mathrm{out}}(T-1)$ and $A_{\mathrm{out}}(T-2)$. Based on the above assumption, three linear predictive models were considered as follows:

Model (1): $\mathrm{In}(T) = f(A_{\mathrm{in}}(T-1), A_{\mathrm{in}}(T-2))$
Model (2): $\mathrm{In}(T) = f(A_{\mathrm{out}}(T-1), A_{\mathrm{out}}(T-2))$
Model (3): $\mathrm{In}(T) = f(A_{\mathrm{in}}(T-1), A_{\mathrm{in}}(T-2), A_{\mathrm{out}}(T-1), A_{\mathrm{out}}(T-2))$

In Figure 1, goodness of fit generated by Models (1), (2) and (3) are compared, with Model (3) has the smallest goodness of fit. The goodness of fit generated by Model (1) is slightly smaller than that generated by Model (1). In order to simplify our approach, we decide to build predictive models by improving Model (1). In Model (1), network traffic load is predicted based on the previous two moving averages. Then, we consider a possibility of improved performance when the model is based on the previous three moving averages. So a new model is proposed as follows:

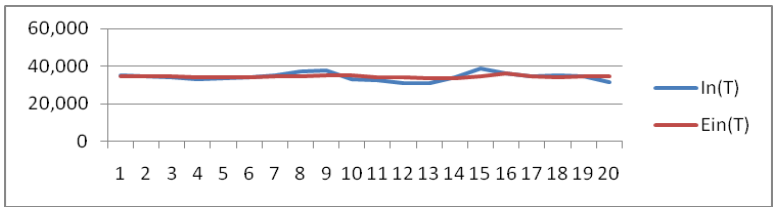
Model (4): $\mathrm{In}(T) = f(A_{\mathrm{in}}(T-1), A_{\mathrm{in}}(T-2), A_{\mathrm{in}}(T-3))$

Goodness of fit generated by models (1) and (4) is compared in Figure 2. It can be seen that Models (4) outperforms Model (1) because of generating smaller goodness of fit.

4.3 Models based on three independent variables

The models described in Section 4.2 have large goodness of fit values. This is due to their poor performance when there are significant oscillations in network load. To deal with these situations, we develop a new approach based on the assumption that using the previous 3 real network load values (three independent variables) for prediction will provide more accurate prediction than using the average network load data. Based on this assumption, Model (5) is proposed:

Fig. 4. Successful prediction



Model (5): $\mathrm{In}(T) = f(\mathrm{In}(T-1), \mathrm{In}(T-2), \mathrm{In}(T-3))$

As is displayed in Fig 3 that Models (5) performs better than Model (4), because Model (5) has a smaller goodness of fit than Model (4).

4.4 Model applied on an hour basis

Although models (5) has the best performance within the above models, the calculated goodness of fit value is still significant. Generally, the increase of goodness of fit value is caused by sudden and dramatic fluctuations in network traffic load. It still needs to be established how well these models work on different type of data (smooth and coarse). When we consider network load data for a whole day, it is more likely that the data will be coarse. However, when we consider the same data in segments that are an hour long, most of the segments will be smooth. Therefore, Model (5) is considered an hour long segments. In this experiment, the first 40-minute data is used to build model, and then the next 20-minute network traffic load is predicted by using the model.

Numerous generated results show that Model (5) provides accurate network load forecasts for certain time periods, however for some other periods forecasts are not sufficiently accurate. Here we only display two sample examples of successful and unsuccessful forecast situations for inbound network load.

In Fig 4 and Fig 5, $E_{\mathrm{in}}(T)$ stands for the estimate value of the inbound network load, and $E_{\mathrm{in}}(T) - \mathrm{In}(T)$ is the deviation of estimated inbound network load from the real inbound network load. The two results based on different data samples (smooth and coarse data respectively) differ dramatically in predicting network load. According to the good prediction, the deviation of inbound network load is very small, with its absolute value varying from 8 KB to 2882 KB. Its maximum deviation 2882 KB is less than 9% of the average inbound load 34,000 KB. In contrast, the maximum deviation of inbound network load in the bad prediction incredibly reaches 57,892 KB, which even surpasses the average inbound network load 40,000 KB.

Since the models work well for the smooth data, as to how to build models based on coarse data and make it work well for prediction would be the next challenge. Besides, this model is not feasible for a continuous prediction, because the prediction of the last 20-minute network load in each hour is based on the model built with the first 40-minute network load, and the first 40-minute network load is not predicted.

Fig. 5. Unsuccessful prediction

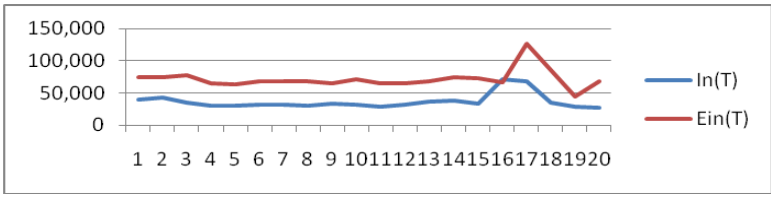
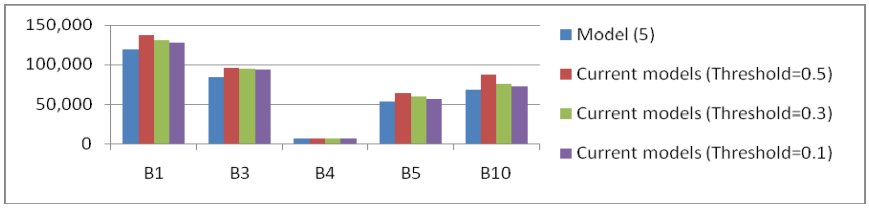


Fig. 6. Goodness of fit obtained from Model (5) and current models



4.5 Continuous predictive models

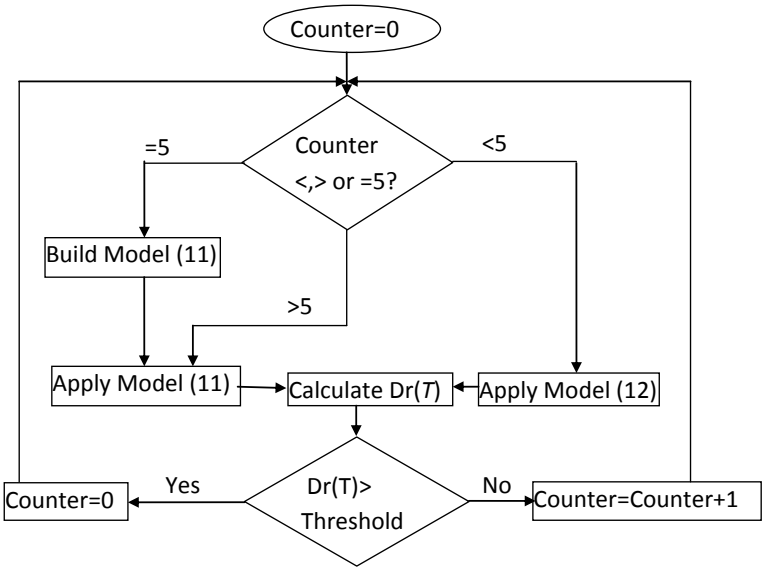
In this section, a predictive model is built by combining Model (5) and one instance of the same model based on constant values (Model (5) - instance A). The weight constants used in Model (5) - instance A are based on results of extensive simulations. The prediction procedure is illustrated in Figure 7 and the combined model is explained as follows:

Model (5): $E_{\text{in}}(T) = f(\text{In}(T-1), \text{In}(T-2), \text{In}(T-3))$
Model (5) instance A: $E_{\text{in}}(T) = 0.85 \times \text{In}(T-1) + 0.1 \times \text{In}(T-2) + 0.05 \times \text{In}(T-3)$
Goodness of fit: $\text{Dr}(T) = \frac{E_{\text{in}}(T-1) - \text{In}(T-1)}{\text{In}(T-1)}$
Goodness of fit ratio threshold: Threshold: (i.e. 0.5, 0.3 and 0.1)

At first, a counter is set to 0. The counter is increased when the goodness of fit calculated for the obtained prediction is within the given threshold. When the goodness of fit is above given threshold than counter is reset to 0. Model (5) instance A is applied to predict network load whenever the counter is smaller than 5. When counter is 5, Model (5) can be built based on the previous 5 network load data and used to predict the next network load. When the counter is greater than 5 already built Model (5) is applied.

As it can be seen from Figure 6, as threshold decreases, the combined model performs better; and when the threshold is equal to 0.1, the goodness of fit generated by the combined model is as small as that generated by purely using Model (5). One advantage of the combined model is that it is very reactive to sudden and dramatic changes in network traffic load. Also, it is easy to be built and implemented without requiring much storage space.

Fig. 7. Algorithm for combining two approaches for the load predicton



5 Discussion

We have described a predictive model that combines local signals, neighbouring signals and signals mined from background information. This paper has focussed on the predictive model for local signals as its solution is more surprising than the other components. A surprising conclusion of these experiments is that a subtle combination of a linear regression based model and a highly reactive model applied to data aggregated at two levels of granularity gives excellent performance. The use of Markov Chains and other non-linear models is not warranted. This result is to an extent a consequence of the simple goal of the local prediction task, that is just to predict whether the load is (roughly) increasing, decreasing or constant. The conclusion is significant as this simple model is easier to built, and requires less computational resources than non-linear methods. More importantly, by flipping between data granularity aggregations it is able to react quickly to sudden and dramatic changes in network traffic load.

The results described in this paper are presently being trialled in simulation experiments at the University of Technology, Sydney where distributed algorithms are being developed for regulating the power consumption of LTE networks. The solution being investigated is a multi-agent system in which an agent is located with each cluster of LTE stations in the network. A key input to support the agent's decision making is the load predications provided by the system described here. Early in 2011 we will conduct trials in the Alcatel-Lucent (Bell Labs) research laboratory located at Blackfriars on the UTS campus.

References

1. Liu, M., Liu, M.T.: A power-saving algorithm combing power management and power control for multihop ieee 802.11 ad hoc networks. International Journal of Ad Hoc and Ubiquitous Computing. **4** (2009) 168–173
2. Sterritt, R., Adamson, K., Shamcott, C., Curran, E.: Data mining telecommunications network data for fault management and development testing. In: Proceedings of Data Mining 2000 Data Mining Methods and Databases for Engineering, Finance and Other Fields, Cambridge, UK, WIT Press (2000) 299–308
3. Sasisekharan, R., Seshadri, V., Weiss, S.M.: Data mining and forecasting in large-scale telecommunication networks. IEEE Intelligent Systems **11** (1996) 37–43
4. Hatonen, K., Klemettinen, M., Mannila, H., Ronkainen, P., Toivonen, H.: Knowledge discovery from telecommunication network alarm databases. In: Proceedings of the Twelfth International Conference, Data Engineering. (1996) 115–122
5. Hermann, W., Klaus-Dieter, T., Klaus, J., Guido, C.: Using neural networks for alarm correlation in cellular phone networks. In: Proceedings of the International Workshop on Applications of Neural Networks in Telecommunications. (1997) 1–10
6. Gardner, R., Harle, D.: Alarm correlation and network fault resolutionusing kohonen self-organising map. In: IEEE Global Telecom Conference, New York, NY, USA, IEEE Computer Society (1997)
7. Gurer, D., Khan, I., Ogier, R.: An artificial intelligence approach to network fault management. Technical report, SRI International, Menlo Park, California, USA. (1996)
8. Weiss, G.: Data mining in telecommunications. In: The Data Mining and Knowledge Discovery Handbook. Springer Verlag (2005) 1189–1201
9. Mika, K., Heikki, M., Hannu, T.: Rule discovery in telecommunication alarm data. Journal of Network and Systems Management **7** (1999) 395–422
10. Weiss, G., Haym, H.: Learning to predict rare events in event sequences. In: Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining, AAAI Press (1998) 1–5
11. Park, B., Kargupta, H.: Distributed data mining. In Ye, N., ed.: Algorithms, Systems, and Applications in Data Mining Handbook. Lawrence Erlbaum Associates (2002) 1–22
12. Chen, T., Liu, S.: A model and evaluation of distributed network management approaches. IEEE Journal on Selected Areas in Communications **20** (2002) 850–857
13. Ogston, E., Vassiliadis, S.: A peer-to-peer agent auction. In Vassiliadis, S., ed.: First International Joint Conference on Automomous Agents and Multi-Agent Systems, New York, NY, USA, ACM (2002) 151–159
14. Kargupta, H., Park, B., Hershberger, D., Johnson, E.: Collective data mining: A new perspective toward distributed data mining. In: Advances in Distributed and Parallel Knowledge Discovery. MIT/AAAI Press (1999) 1–38
15. Kargupta, H., Hamzaoglu, I., Stafford, B.: Scalable, distributed data mining using an agent based architecture. In: International Conference on Knowledge Discovery and Data Mining, Newport Beach, California, USA (1997) 211–214
16. Grossman, R., Sivakumar, H., Turinsky, A.: Papyrus: a system for data mining over local and wide area clusters and super-clusters. In: Proceedings of the 1999 ACM/IEEE conference on Supercomputing. ACM Press (1999) 1–8
17. Bandyopadhyay, S., Giannella, C., Maulik, U., Kargupta, H., Liu, K., Datta, S.: Clustering distributed data streams in peer-to-peer environments. Information Sciences **176** (2006) 1952–1985
18. Roughan, M., Zhang, Y.: Secure distributed data-mining and its application to large-scale network measurements. ACM SIGCOMM Computer Communication Review. **36** (2006) 7–14