Sensors and Materials, Vol. 33, No. 4 (2021) 1353–1361
MYU Tokyo

S & M 2539

Network Flow Queuing Delay Prediction for City Public Services Based on Long Short-term Memory

Long Zhang,[1] Yu Chen,[2*] Xinyi Huang,[3] Cheng-Fu Yang,[4,5**] and Peng Xue[1]

[1]College of Mathematics and Information Engineering, Longyan University, Fujian 364012, China
[2]Fuzhou Institute of Technology, Fuzhou, Fujian 350506, China
[3]College of Oral Medicine, Shandong University, Jinan 250100, China
[4]Department of Chemical and Materials Engineering, National University of Kaohsiung, Kaohsiung 811, Taiwan
[5]Department of Aeronautical Engineering, Chaoyang University of Technology, Taichung 413, Taiwan

(Received October 21, 2020; accepted January 28, 2021)

Keywords: load prediction, Spring Cloud, microservice architecture, network flow queuing delay, long short-term memory (LSTM)

It is very important to accurately predict the network flow queuing delay to improve the network performance of city public services. City public services are the offspring of the paradigm "smart + connected communities" and aim to overcome the problems of isolated and fragile data collection because of administrative divisions. Quality of service is one of the important evaluation indexes in service-level agreements, in which low delay is a basic requirement for measurement. To improve city public services and predict the network flow queuing delay of city public services in advance, we propose a framework based on long short-term memory (LSTM) that will allow the government to enhance service efficiencies and offer better service experiences for every citizen. The results obtained by using the investigated framework show that the proposed algorithm has superior aggregated prediction accuracy and real-time performance to other methods.

1. Introduction

In computer and telecommunication engineering, there is a queuing delay when the required processing data waits in a series or cascade until it can be executed, which is a very important delay in signal transmission networks. In a switched network, a queuing delay exists between the completion of the originating signal and the arrival of the ringing signal at the receiver. There are many causes of the queuing delay, including the delay at the originating switches, at the intermediate switches, or at the servicing switches of call receivers. In a data network, the queuing delay is defined as the sum of the delays between the services' requests and the establishment of the circuits in the data's terminal equipment. With the fast transmission of the 4G system and the high transmission efficiency of the 5G system in communication technologies and the dramatic improvements in artificial intelligence and microchips, we have witnessed many creations and innovations such as popular mobile applications including TikTok, futuristic IoT deployments, and various types of cloud services.(1) City public services, referred to collectively as a Smart City, are the offspring of the paradigm "smart + connected

*Corresponding author: e-mail: chen_fj@qq.com
**Corresponding author: e-mail: cfyang@nuk.edu.tw
https://doi.org/10.18494/SAM.2021.3173

ISSN 0914-4935

communities". This system was set up to overcome the problems of isolated and fragile data connection because of administrative divisions, so that citizens can share big-data resources and recreate new and valuable resources for themselves by moving public services from offline and PCs to mobile ends. As a result, the government can enhance service efficiencies and offer better service experiences for every citizen.(2)

Traditional platforms of city public services have often adopted the "Spring + SpringMVC + MyBatis" (SSM) framework as their general solution. All the SSM frameworks play a part in a software background system and are combined into independent units, such as war or jar packages, and these units are deployed in a Docker such as Tomcat, which is called a solo application architecture.(3) These systems are difficult for government to maintain and upgrade when there is high demand for public services. Thus, a microservice architecture is selected for ultrascale access schemes, where different functions are categorized and divided into independent microservices, all of them running in independent processes with their own exclusive database.(4) Quality of service is one of the important evaluation indexes in service-level agreements, in which low delay is a basic measurement. There are four different types of delay, namely, process delay, transmission delay, propagation delay, and queuing delay, the last being the most critical one.(5)

As network congestion is inevitable, investigating the possibility of increasing the bandwidth has limited effectiveness in enhancing service performances because it is an uneconomical method. This makes research on delay prediction very important. We propose a new and improved long short-term memory (LSTM) framework that uses multiple scales to forecast the network traffic forecast on an operating system. We use the recurrent neural network (RNN) to propose an improved version of LSTM based on the Spring Cloud system and use this system to predict the network traffic of city public services. We use the RNN-improved multiscale LSTM framework as a specific method and study its actual application, and we also compare its prediction results with those of other similar algorithms. The proposed framework can be applied in special cases, for example, a special holiday module can be added for the quantitative analysis of network traffic.

2. Related Research

The framework of a traffic prediction model can be classified into two different types, linear and nonlinear. Both the autoregressive integrated moving average (ARIMA) and the Kalman filter are linear models. The requisites of the network flow are that it must remain constant, stable, and linear, although it is difficult to satisfy these requirements.(6) Nonlinear models include grey prediction, wavelet transform prediction, support vector regression (SVR) prediction, and neural network prediction. Joshi and Hadi compared these prediction models of neural networks based on an artificial neural network (ANN) with other methods, and he proved that the ANN model is usually better than other models.(7) Chen *et al.* proposed a flow prediction model of short-time traffic using a back-propagation ANN for a two-lane undivided highway.(8) The use of RNNs has recently become increasingly popular in the application of time-series prediction, and Zagrebina used an RNN to investigate effective electricity prediction.(9) LSTM is a variant type of RNN that continues learning and overcoming vanishing

gradient problems for a long time using its unique selective memory, making it very suitable for network flow prediction involving time series with long correlations.[10] LSTM can process not only single data points (such as images), but also entire sequences of data. In this study, we use LSTM technology to predict the network flow queuing delay of city public services as an example of multivariate data analysis. A remote microwave sensor can be used to capture nonlinear dynamic data in systems providing city public services, and LSTM can be used in combination with a neural network for traffic speed prediction. Chen *et al.* investigated attention-based bidirectional long short-term memory (ABLSTM) for passive human activity recognition using WiFi channel state information signals.[11] For this purpose, we propose an RNN-improved multiscale LSTM framework as a specific method and apply it to the delay prediction of network traffic for city public services.

3. City Public Services Architecture

City public services, which have been transformed from a past SSM architecture to a microservice architecture, can be accessed from PCs and mobile ends, including iOS and Android systems, and they can satisfy the various requirements of citizens in everyday straightforward government procedures, as shown in Fig. 1. City public services include identity

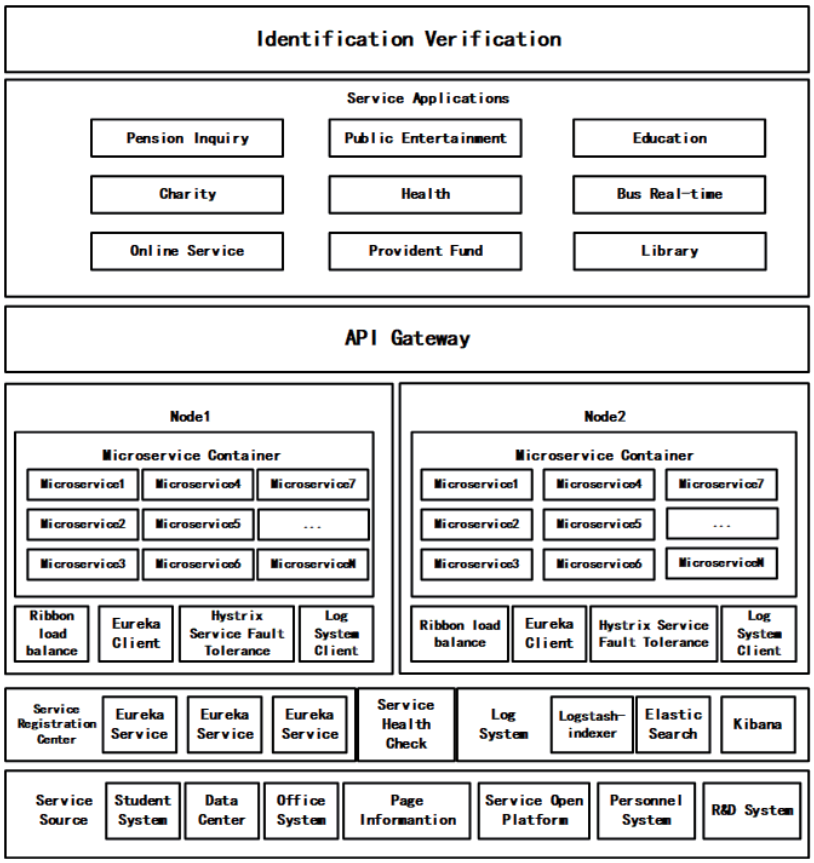

Fig. 1. City public services architecture.

confirmation, public transport, social insurance, real estate, public welfare, and education, where many modules are directly linked to corresponding administrative bureaus. Thus, city public services are very important in supporting various services. These services have been developed using different technologies and teams without much intra-service communication, suggesting that reducing the coupling in software engineering is important.

Spring Cloud, Spring Boot, and MyBatis are adopted in our microservice architecture of city public services, in which Spring Cloud is the microservice framework and is responsible for global service governance. Spring Boot acts like a package of rapid configuration scaffolding, which is dedicated to developing solo applications.[12] The Eureka server is used for service registration and discovery, and every newly deployed service must be registered at Eureka. Zuul is an L7 application gateway that acts as a service gateway and offers functions such as routing and filtering with all external requests to be verified before entering. Config provides centralized configuration management as there are many microservices, each with its own configuration files. Ribbon, which is critical as it offers load balance, is deployed in Zuul and supports third-party developments, in which we can deploy our network flow queuing delay prediction method.

4. Experiments

4.1 LSTM

An LSTM system, which can act as an upgrade model of an RNN, can handle long correlation time series by persistent learning and can avoid gradient vanishing based on selective memory. It also has a good prediction performance for network flow queuing delay. The memory cell is a critical sub-component in LSTM, where there is a self-cycling connection neuron connected with three related gates: the input gate, output gate, and forgotten gate. Self-cycling connections can maintain memory cells in a stable state between two time stamps without external interference. Data are processed by the activation function in the input gate, and every unit is integrated with the output gate to calculate the corresponding time series value. The forgotten gate is used for data optimization, which can select the best time gap for an input series by adjusting self-cycling connections through remembering or leaving past states with the application of the LSTM unit structure, as shown in Fig. 2.[13,14]

Assuming the input series is expressed as $X = (x_1, x_2, ..., x_n)$ at each moment, the output series is defined as $Y = (y_1, y_2, ..., y_n)$, and each state in the invisible layer is defined as $H = (h_1, h_2, ..., h_n)$, then the prediction procedures based on LSTM can be described as follows:

The output of the input gate is defined as

$$input_{gate_t} = \sigma(w_{ix}x_t + w_{hh}h_{t-1} + w_{is}s_{t-1} + b_i). \quad (1)$$

The output of the forgotten gate is defined as

$$forgotten_{gate_t} = \sigma(w_{fx}x_t + w_{hh}h_{t-1} + w_{fs}s_{t-1} + b_f). \quad (2)$$

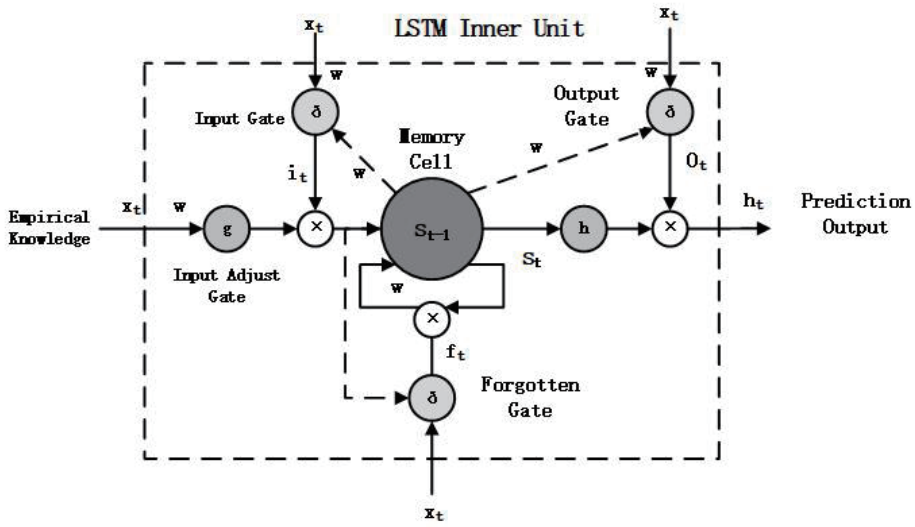

Fig. 2. LSTM unit structure.

Then, the memory unit can be updated as

$$s_t = forgotten_{gate_t} \circ s_{t-1} + input_{gate_t} \circ g(w_{sx}x_t + w_{hh}x_{t-1} + w_{ss}s_{t-1} + b_s). \tag{3}$$

The output of the output gate is defined as

$$output\_gate_t = \sigma(w_{ox}x_t + w_{hh}h_{t-1} + w_{os}s_{t-1} + b_o). \tag{4}$$

The prediction output value is defined as

$$p_t = output\_gate_t \circ p(s_t). \tag{5}$$

In the above equations, σ and g are activation functions, b is the offset, and o is the Hadamard product operator. The former term in Eq. (3) decides whether to keep or leave the parts of information in state s_{t-1}, while the latter term chooses the remaining input data. After Eq. (4) is used to calculate the output of the next invisible layer, the final prediction value is obtained from Eq. (5) with convergence and update in the calculation of the whole memory unit.

4.2 Multiscale integration prediction model based on LSTM framework

The created network flow of city public services originates from citizens with different purposes involved in various events. This flow is different from the network flow applied in videos and games, which constantly remains stable and is easy to predict. Many microservice modules can be used in the government's public service platform, each with a unique service that creates a designated network flow. For example, the education service is usually congested in June and September and the pension fund is busy in November. This means that prediction

models based on small-scale events can hardly overcome these special circumstances, which is also called the affordability problem. Thus, we sample and segment the data in different scales as the input of different cascaded LSTM frameworks, where external factors such as weather, important holidays, and special events are considered as different and special time periods, then these parameters can be added into the database to improve the framework's accuracy. The model for prediction in our study is presented in Fig. 3.

In Fig. 3, $S_{fundamental}$ is the feature abstraction sampling based on a unit interval, S_{scale1} is the feature abstraction sampling on n intervals, S_{scale2}, S_{scale3}, ..., S_{scaleN} are supported by similar operations, and $S_{extension}$ contains special data, such as holiday information, important public affairs, and dramatic changes in the weather, and is the output after the process in the fully connected layer involving one-shot encoding. In general, the cascaded LSTM framework attempts to extract data features in different scales with the fully connected layer while dealing with the special data mentioned above. In the merge layer, different fetched tensors are assigned as designated weight values and are adjusted to their optimal conditions during the training phases. Thus, we can obtain the output from the following equation:

$$S_{out} = w_e \circ S_{extension} + w_N \circ S_{scaleN} + \ldots + w_1 \circ S_{scale1} + w_{fundamental} \circ S_{fundamental}. \tag{6}$$

The loss function adopted in our research is evaluated using the mean square error (MSE), given by Eq. (7). The L2 loss (quadratic loss) is measured using the Euclidean distance, which is suitable for the network flow here because there are many random components and anomalous points that cannot be ignored.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \overline{Y}_i\right)^2 \tag{7}$$

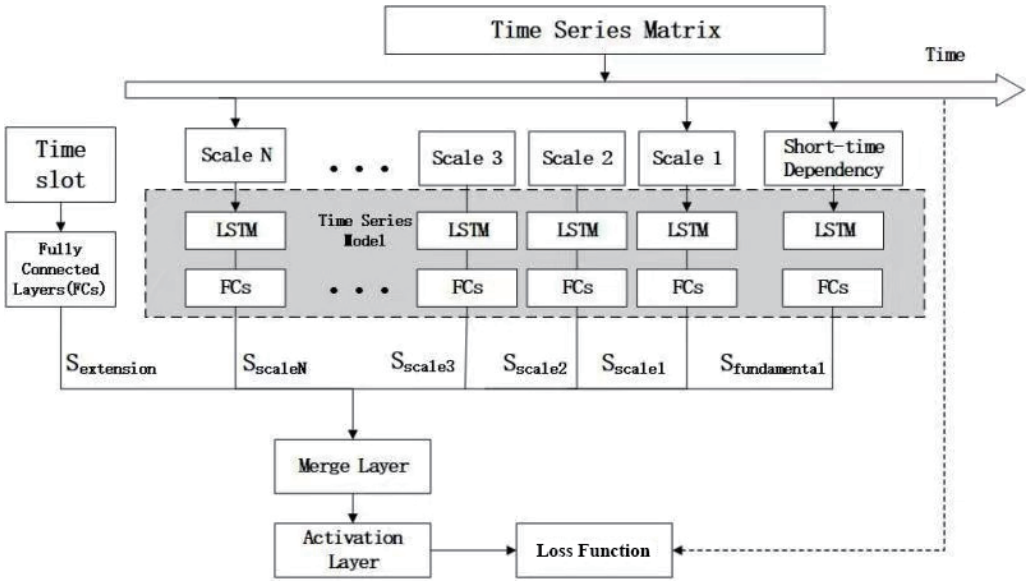

Fig. 3. Structure for integration of multiscale prediction model.

Both the activation functions in our framework are nonlinear and can be expressed with two involved activation methods, the sigmoid for the memory cell's input activation and tanh for the fully connected layer, respectively expressed by Eqs. (8) and. (9).

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{8}$$

$$\tanh(z) = g = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} \tag{9}$$

4.3 Experimental results

To verify the effectiveness and accuracy of our multiscale integration prediction model, we compared the performances of the training model with the optimal parameters and without integration, and the training model with the optimal parameters and with two-scale and three-scale integrations. The relative parameters and results are shown in Tables 1–3, where we use the optimal training parameters. The traditional model (without any integration) only uses the most recent dependence for predictions, with the scope of their predictions based on the minimum sampling interval, which lasts 200 ms. The parameter back1 refers to the number of timestamps of forward feeding, and the parameters back2 and back3 have similar meanings. Dropout is applied between different LSTM frameworks, and the regularization item of the density of the fully connected layer is added to the weight values.

Through the experiment results, we find that the LSTM framework with three-scale integration has the best prediction accuracy but sacrifices real-time performance owing to large-scale overlapping. We compare the two values of LSTM and LSTM with three-scale integration (LSTM-MERGE) with other representative frameworks,[15] and then we record their performances for different prediction scopes, as measured by the root mean square error (RMSE).

Table 1
Training model with optimal parameters without integration.

back1	LSTM	Dropout	Density	Regularization coefficient	Training MSE	Test RMSE
5	25, 25	0.1	18, 15	0.02	152.38	154.27

Table 2
Training model with optimal parameters and two-scale integration.

back1 back2	interval	LSTM	Dropout	Density	Regularization coefficient	Training MSE	Test RMSE
3, 3	2	25	0.1	18, 15	0.03	149.67	151.23

Table 3
Training model with optimal parameters and three-scale integration.

back1 back2 back3	interval1 interval2	LSTM	Dropout	Density	Regularization coefficient	Training MSE	Test RMSE
3, 3, 3	2, 4	25	0.1	18, 15	0.03	142.29	145.78

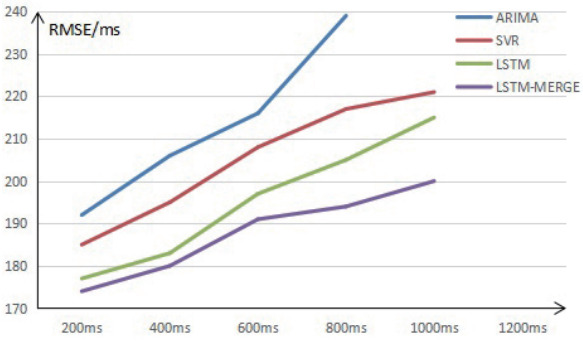

Fig. 4. (Color online) Comparison of RMSE values for different methods and intervals.

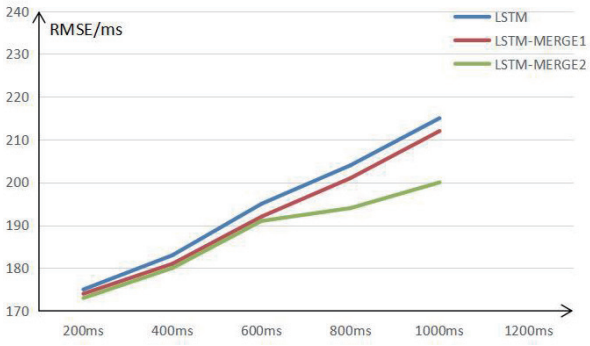

Fig. 5. (Color online) Comparison of RMSE among different LSTMs.

The RMSE of different methods (ARIMA, SVR, LSTM, and LSTM-MERGE) and intervals are shown in Fig. 4, in which we extend the prediction scope to equal time sampling intervals of the most recent dependence, and 200, 400, 600, 800, and 1000 ms are chosen as the intervals.

It is inferred that the ARIMA model, which is a parameter model, has the least accuracy among the methods, which means that it cannot compete with machine learning methods. The ARIMA model can fit the time series data to predict future points in the series (forecasting), and it is applied in some cases where data shows evidence of non-stationarity in the sense of the mean. The SVR algorithm is the most common application form of support vector machines (SVMs). The SVR algorithm can successfully estimate the load ability margin under normal operating conditions and different loading directions, and it has the feature of minimizing the error when achieving a generalized network. However, we propose the RNN-improved multiscale LSTM framework, and we compare its prediction results with those of ARIMA, SVR, and single-scale LSTM. The performance of all models deteriorates when the prediction scope increases, and the LSTM based on three-scale integration has the slowest increase in RMSE. However, as shown in Fig. 4, for the same sampling interval, the RMSE values of the investigated LSTM-MERGE framework are lower than those of the ARIMA, SVR, and LSTM methods. To test the effectiveness of LSTM with three-scale integration, we carry out another test to compare the performances of LSTM, LSTM with two-scale integration (LSTM-MERGE1), and LSTM with three-scale integration (LSTM-MERGE2), and the results are given in Fig. 5. All methods showed lower performance with increasing prediction interval but the RMSE of LSTM-MERGE2 increased most slowly, consistent with the previous test results.

5. Conclusion

We investigated an improved network flow queuing delay prediction method based on LSTM with multiscale integration, which is applied to city public services based on a microservice architecture. The investigated system has a large number of different microservice modules. As compared with the ARIMA model, the SVR model, and the LSTM framework, we found that LSTM with a multiscale integration prediction framework had superior aggregated prediction accuracy and real-time performance.

Acknowledgments

This work was supported by project numbers MOST 109-2622-E-390-001-CC3 and MOST 109-2221-E-390-023 and the Young and Middle-aged Teacher Program of the Education Department of Fujian (JAT200601).

References

1 H. S. Yang and Y. H. Kim: Sensors **19** (2019) 3276.
2 L. Ruan, C. Y. Li, Y. Zhang, and H. X. Wang: Comput. Environ. Urban Syst. **77** (2019) 101268.
3 B. Bergvall-Kareborn: Syst. Pract. Action Res. **15** (2002) 309.
4 F. Rademacher, J. Sorgalla, and S. Sachweh: IEEE Software **35** (2018) 36.
5 N. D. Adesh and A. Renuka A: Comput. Commun. **146** (1029) 131.
6 X. Ning, Y. Dang, and Y. Gong: Energy **118** (2017) 473.
7 M. R. Joshi and T. H. Hadi: A Review of Network Traffic Analysis and Prediction Techniques, School of Computer Sciences, North Maharashtra University (2015).
8 Z. Zhao, W. Chen, X. Wu, P. C. Y. Chen, and J. Liu: IET Intell. Transp. Syst. **11** (2017) 66.
9 S. A. Zagrebina, V. G. Mokhov, and V. I. Tsimbol: Procedia Comput. Sci. **150** (2019) 340.
10 K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink, and J. Schmidhuber: IEEE Trans. Neural Networks Learn. Syst. **28** (2016) 2222.
11 Z. Chen, L. Zhang, C. Jiang, Z. Cao, and W. Cui: IEEE Trans. Mobile Comput. **1** (2019) 2714.
12 N. Tufek, M. Yalcin, M. Altintas, F. Kalaoglu, Y. Li, and S. K. Bahadir: IEEE Sens. J. **20** (2020) 3101.
13 X. L. Ma, Z. M. Tao, Y. H. Wang, H. Y. Yu, and Y. P. Wang: Transp. Res. Part C: Emerging Technol. **54** (2015) 187.
14 Z. Zhao, W. H. Chen, X. M. Wu, P. C. Y. Chen, and J. M. Liu: IET Intel. Trans. Syst. **11** (2017) 68.
15 B. M. Williams and L. A. Hoel: J. Trans. Eng. **129** (2003) 664.