ELECTRONICS AND ELECTRICAL ENGINEERING
ISSN 1392 – 1215 2009. No. 4(92)
ELEKTRONIKA IR ELEKTROTECHNIKA

TELECOMMUNICATIONS ENGINEERING
T180
TELEKOMUNIKACIJŲ INŽINERIJA

Metamodelling of Queuing Systems using Fuzzy Graphs

G. Lauks, J. Jeļinskis
Institute of Telecommunication, Faculty of Electronics and Telecommunication, Riga Technical University
Āzenes st. 12, LV-51368 Riga, Latvia, phone: +371 7089201; e-mail: lauks@rsf.rtu.lv, jans.jelinskis@etf.rtu.lv

Introduction

The dynamic behavior of queuing systems with complicated traffic can be analyzed using simulation models. Unfortunately, often the results are only available in a form of large datasets, which makes it hard to extract the underlying regularities (knowledge). Additionally, each simulation run is time consuming and interactive analysis is usually impossible. One solution is to approximate the behavior of simulation model, resulting in a metamodel as a model of an input/output function. There are some approaches: design of experiments methods, Latin hypercube sampling design, application driven sequential design, simulation metamodelling [5] and others.

Metamodels of an actual queuing system exactly have a large set of variables (high dimensions). It includes the topologies of Networks (systems), probability distributions of service time and inter-arrival time, routing strategies, load and state dependent strategies of current queue length, performance indices such as throughput, queue length, residence time, utilization and etc. Simulations of actual queuing systems with large amount of dimensions consumes inexpiable amount of resources and gives the non precise results. In addition, the every variable dimension takes the small number of simulation attempts. All this leads us to the fuzzy systems.

On the other hand automatic extraction of rules from Network on-line measurements using machine learning algorithms promises the possibility to construct the knowledge base for decision making in the real time conditions. The goal of the current research is to approximate the large datasets of measurements or the simulations of queuing systems using Fuzzy Graphs and Rectangle Basis Function Neural Networks using them as a machine learning mechanisms. Fig. 1 represents the Fuzzy Graph Rectangular Basis Function Network diagram.

RecBFN is a variation of RBF networks, in which the representation is a set of hyper-rectangles belonging to the classes of the system. Every dimension of the hyper-rectangles represents a membership function (MF).

A set of hyper-rectangles represents rules and knowledge. The Radial Basis Functions in RBS neural networks can be interpreted independently. But the most desired form of rules for applications in the knowledge basis is the following: IF $\{(x_1 \in [x_1^a, x_1^b]) \cap ... \cap (x_n \in [x_n^a, x_n^b])\}$, THEN Class C, where x_i^a, x_i^b is lower and upper bound of the interval corresponding to the i-th attribute.

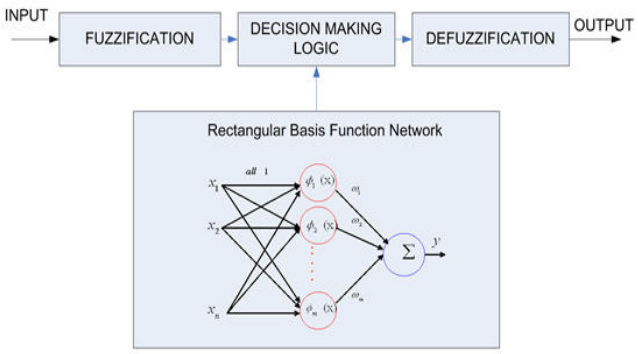

Fig. 1. The Fuzzy Graph Rectangular Basis Function Network diagram

It corresponds with ideas of granular knowledge structure and granular computing. Granulation is partitioning (crisp or fuzzy) of an object into a collection of granules, with a granule being a clump of elements drawn together by indistinguishability, equivalence, similarity, proximity or functionality.

For example, traditionally in queuing networks our interest lies to three classes of systems: under-loaded, loaded and overloaded. Inside every one of them all the systems are indistinguishable. It means that we do not need to calculate precisely the characteristic parameters such as the Hurst coefficient.

Fuzzy Graph Rectangular Basis Function network

Rectangular Basis Functions Networks (RecBFN) comes from RBF Networks and were presented by M. Berthold and K.P. Huber in [1, 2]. They refer to a system that is able to obtain a fuzzy model from training data, by

means of the execution of the modified Dynamic Decay Adjustment (DDA) algorithm.

The Fuzzy Graph RecBFN offers a promising way to build a metamodel from simulation data that allows analyzing the underlying system behavior. Every neuron is represented by a Fuzzy Point (FP).

The following definitions are used in this paper:

Each RecBFN of prototype p_i^c of class c with index i. It consists of:

- An activation function $R_i^c(.)$.
- A reference vector (center): $\vec{r}_i^c = (....)$.
- Weight: A_i^c.
- Two sets of "radii".
  - Set of axis along which the rectangle which is spread out towards infinity.
  - Set of axis along which the rectangle which is restricted with the radius of σ_j.
- Number of rules for class c: m_c.

The most desired form of rules is: IF $\{(x_1 \in [x_1^a, x_1^b]) \cap ... \cap (x_n \in [x_n^a, x_n^b])\}$, THEN Class C, where x_i^a, x_i^b is a lower and upper bound of the interval corresponding to the i-th attribute. Activation function:

$$A(x_i, r_i, \sigma_i) = \begin{cases} 1 : |x_i - r_i| \le \sigma_i \\ 0 : else..... \end{cases} \quad (1)$$

Training

Training of the Network requires classified input patterns. Therefore the fuzzification unit is used to determine which class each pattern belongs to.

The Network is trained to correctly classify the corresponding input patterns.

Function approximation takes the reverse step , the Network produces membership values for all classes and the defuzzification unit computes the final output value using well-known center-of-gravity method. Training of RecBFN is done using DDA.

The algorithm from [8] is based on three steps that introduce new RecBF when necessary and adjusts the core and support regions of existing RecBFs:

Covered: if a new training pattern lies inside the support region of an already existing RecBF with correct class its core region is extended to cover a new pattern;

Commit: if a new pattern is not covered by a RecBF of the correct class a new hidden unit will be introduced. Its reference vector will be the same as the new training instance and the widths will be chosen as large as possible, without running into conflict with already existing prototype of incompatible classes.

Shrink: if a new pattern is incorrectly covered by an already existing RecBF of an incompatible class, this prototype`s support area will be reduced (e.g. shrunk) so that conflict is solved.

The program of DDA is presented in [6]. The algorithm of the DDA program is depicted below:

```
START
    DDA-RecBFN
//resetting weights
        FORALL prototypes p_i^c DO
            A_i^x = 0.0
        ENDFOR
//training one epoch
        FORALL training patterns (x⃗, c) DO
            IF ∃p_i^c : R_i^c(x⃗) = 1 THEN
                A_i^c += 1.0
            ELSE
            // "commit" -new prototype p_{m_c+1}^c with r⃗_{m_c+1}^c
                FORALL 1 ≤ i ≤ n DO
                    σ_i = ∞
                FORALL k ≠ c, 1 ≤ j ≤ m_k DO
                    p_{m_c+1}^c → SHRINK(p_j^k)
                    A_{m_c+1}^c = 1.0
                    m_c += 1
            ENDIF
            //"shrink": adjust conflicting prototypes
            FORALL k ≠ c, 1 ≤ j ≤ m_k DO
                p_j^k → SHRINK(x⃗)
            ENDFOR
END
```

The DDA program algorithm is depicted above, and below is given the algorithm for Shrink method used in the DDA algorithm.

```
METHOD shrink
    IF σ_best does not exist THEN
        σ_k = σ_best,k
    ELSEIF σ_max,i ≥ σ_best,j
        σ_i = σ_max,i
    ELSE
        σ_j = σ_min,j
    ENDIF
END METHOD
```

Experiments

The data sets used for training and validation purposes were obtained by discrete-event simulation with JMT [6] and selected such that they cover the distinguishable traffic patterns. We demonstrate the impact of Self-Similarity on Mean Queue Length as a function of link buffer size. We use a single OnOff source with Pareto distribution of interarrival time and parameter α as a variable. The goal of this research is the approximation of mean queue length as a function of buffer size and parameter α using Fuzzy Graphs.

The proposed method was applied to small data sets of simulation experiment of simple Pareto/M/1 queue. In this research we show how the RecBF Network can be used to approximate well known Pareto/M/1 system with 100 simulation points.

We use the following notation:

- BS - Link buffer size (kB)
- Pareto distribution parameters: $1 < \alpha < 2$.
- MQL- Mean queue length in bytes.
- Self-similarity with classes: non-significant, low significance, medium significance, high significance;

The Fuzzy Graph is presented in Table 1. Table 1 shows the resulting function approximation by Fuzzy Graph RecBFN. Number of training epochs -3.

Table 1. The Fuzzy Graph

IF		THEN
$BS \in [10;130]$ $MQL \in [1000;5000]$	AND	non-significant $\alpha \in [2;5]$
$BS \in [10;130]$ $MQL \in [1000;6000]$	AND	low significance $\alpha \in [1,6;1,7]$
$BS \in [30;130]$ $MQL \in [5000,15000]$	AND	medium significance $\alpha \in [1,3;1,4]$
$BS \in [50,130]$ $MQL \in [15000,35000]$	AND	high significance $\alpha \in [1,0;1,1]$

Table 1 shows the impact of increasing self-similarity (by reducing the Pareto distribution's parameter, α on the mean queue length.

Discussion

As an example of some specific applications is the management of the bottleneck links, which are basically caused by routers, switches or similar elements in the networks topology. For this kind of elements some mechanisms, policies and disciplines are of interest. The examples of these are: queue management (RED, BLUE, REM, etc.), schedulers (WFQ,GPS,..), markers, shapers, etc.

The knowledge base developed in such a way presented in this investigation is useful for managing the quality of service architectures such as the DiffServ. The fast and efficient method which uses Fuzzy Graph for metamodel representation, means that the rules can be extracted in the manner which is compact enough for the decision making process can be done even in the on-line mode, with no need for complicated and precise computing.

Conclusions and future research

Construction of the Fuzzy Graph is done in a fast and efficient way without any need for user interaction, besides the definition of the output fuzzification. It means that such an approach is an attractive tool for on-line decision support systems management and knowledge basis establishment.

The Fuzzy Graph RecBFN offers a promising way to build a metamodel from simulation data of Queue Networks (systems) that allows analyzing the underlying system behavior. In addition the extracted Fuzzy Graph is an understandable representation of the knowledge acquired by the Neural Network.

The future research in this field will consider into account the possible aspects of differentiation of the metamodel application. In fact, every communication system behavior which is based on many parameters and is continuously changing need can be simulated and metamodelling can significantly reduce often so unnecessary precise representation of system model. The metamodelling can be the base of the applications in the fast and robust communication system control.

References

1. **Berthold M. R., Huber K. P.** Neural Network based Construction of Fuzzy Graphs.
2. **Berthold M. R., Huber K. P.** From Radial to Rectangular Basis Functions: A new Approach for Rule Learning from Large Datasets.
3. **Soler V., Roig J., Prim M.** Fuzzy Rule Extraction Using Recombined RecBF for Very-Imbalanced Datasets,
4. **Soler V., Prim M.** Rectangular Basis Functions Applied to Imbalanced Datasets.
5. **Merkurjeva G., Napalkova L.** Applications of Neurofuzzy training algorithms to simulation metamodeling.
6. **Bertoli M., Casale G., Serazzi G.** The JMT Simulator for Performance Evaluation of Non-Product-Form // Queueing Networks, Simulation Symposium, ANSS 40th Annual. – 2007. – P. 3–10.
7. **Berthold M. R., Diamond J.** Boosting the performance of RBF networks with dynamic decay adjustment // Advances in Neural Information Processing Systems. MIT press. – 1995. – P. 521–528.
8. **Berthold M. R., Huber K. P.** Neural Network based Construction of Fuzzy Graphs // Second Annual Joint Conference on Information Sciences. North Carolina, USA. – 1995.

Received 2009 02 17

G. Lauks, J. Jeļinskis. Metamodelling of Queuing Systems using Fuzzy Graphs // Electronics and Electrical Engineering. – Kaunas: Technologija, 2009. – No. 4(92). – P. 61–64.

The dynamic behavior of queuing systems under sophisticated traffic can be analyzed using simulation models. Unfortunately, often the results are only available in a form of large datasets, which makes it hard to extract the underlying regularities. One of the interesting applications is the approximation of the behavior of simulation models, called metamodelling. The goal of this paper is to approximate the behavior of queuing systems as well as to extract some understandable knowledge about the simulation model. In this paper we present the knowledge extraction from trained Neural Networks. The underlying knowledge can be extracted from the Network in form of a Fuzzy Graph. The Fuzzy Graphs are generated using Rectangular Basis Functions of Neural Networks. The research results are illustrated with a range of experiments performed. Ill. 1, bibl. 8 (in English; summaries in English, Russian and Lithuanian).

Г. Лаукс, Я. Елинскис. Метамоделирование систем массового обслуживания с использованием нечетких графов // Электроника и электротехника. – Каунас: Технология, 2009. – № 4(92). – С. 61–64.

Динамическое поведение очереди системы может быть проанализировано с использованием имитационных моделей. К сожалению, нередко результаты доступны только в виде крупных наборов данных, из которых трудно извлечь основные закономерности. Одним из интересных приложений приближенного поведения моделей является метамоделирование. Цель исследования состоит в том, чтобы приблизить поведение очередей системы и получить некоторые знания о модели. В этой

статье представлены результаты извлечения знаний из обученных нейронных сетей. Основные знания могут быть получены из сети в виде нечетких графов. Нечеткие графы создаются с использованием прямоугольных базисных функций нейронных сетей. Результаты исследования проиллюстрированы с несколькими экспериментами. Ил. 1, библ. 8 (на английском языке; рефераты на английском, русском и литовском яз.).

G. Lauks, J. Jeļinskis. Metasistemų modeliavimas taikant neraiškius grafus // Elektronika ir elektrotechnika. – Kaunas: Technologija, 2009. – Nr. 4(92). – P. 61–64.

Atliktas dinaminis sudėtingų sistemų elgsenos modeliavimas taikant atskirus modelius. Deja, dažnai tyrimų rezultatai yra pateikiami tik didelių duomenų rinkinių pavidalu, todėl sunku išskirti reikiamus rezultatus. Vienas iš galimų sprendimo būdų – taikyti modeliuose elgsenos aproksimaciją, t. y. metamodeliavimą. Šiame darbe ištirta, kaip išskirti žinias iš apmokytų neuroninių tinklų. Pagrindinės žinios iš tinklų išskiriamos neraiškiųjų grafų pavidalu. Neraiškieji grafai suformuojami pagal neuroninių tinklų pagrindines stačiakampes funkcijas. Il. 1, bibl. 8 (anglų kalba; santraukos anglų, rusų ir lietuvių k.).