ELSEVIER

Advanced Engineering Informatics 16 (2002) 241–246

ADVANCED ENGINEERING INFORMATICS

www.elsevier.com/locate/aei

Artificial neural networks for job shop simulation

Daniel J. Fonseca*, Daniel Navaresse

Department of Industrial Engineering, The University of Alabama, 233 Mineral Industries Building, Tuscaloosa, AL 35487 0288, USA

Abstract

This paper explores the use of artificial neural networks (ANNs) as a valid alternative to the traditional job-shop simulation approach. Feed forward, multi-layered neural network metamodels were trained through the back-error-propagation (BEP) learning algorithm to provide a versatile job-shop scheduling analysis framework. The constructed neural network architectures were capable of satisfactorily estimating the manufacturing lead times (MLT) for orders simultaneously processed in a four-machine job shop. The MLTs produced by the developed ANN models turned out to be as valid as the data generated from three well-known simulation packages, i.e. Arena, SIMAN, and ProModel. The ANN outputs proved not to be substantially different from the results provided by other valid models such as SIMAN and ProModel when compared against the adopted baseline, Arena. The ANN-based simulations were able to fairly capture the underlying relationship between jobs' machine sequences and their resulting average flowtimes, which proves that ANNs are a viable tool for stochastic simulation metamodeling.

Keywords: Artificial neural networks; Metamodeling; Job shop simulation; Manufacturing lead time

1. Introduction

Scheduling has been defined as 'the art of assigning resources to tasks in order to insure the termination of these tasks in a reasonable amount of time' [3]. The general problem is to find a sequence in which the jobs (e.g. basic tasks) pass through the resources (e.g. machines), which constitutes a feasible and optimal schedule with respect to some specific performance criterion [7]. This represents the last stage of the planning activities before production takes place.

Systems simulation has become a powerful decision-making instrument for job shop scheduling. It requires a few simplifying assumptions, captures many of the true characteristics of the real model, and provides good insights about the interactions and relationships between qualitative and quantitative variables. However, a major shortcoming of simulation is the need for expert assistance any time a change is required in a model [14,16].

One contribution to the simplification of the scheduling decision-making process might consist of the development of a system that can perform a rapid evaluation of different alternatives, without the necessity of computer simulation expertise. If a dynamic model of a system could be constructed and presented to the analyst as a black box, one of the major drawbacks of systems simulation would be overcome: the need for a human expert to carry out the simulation.

Artificial intelligence (AI) is the generic name given to the field of computer science dedicated to the development of programs that attempt to replicate human intelligence. Artificial neural networks (ANNs) is one of the AI techniques that has gained an important role in solving problems with extreme difficult or unknown analytical solutions [12]. An ANN consists of an interconnected web of special units, called *neurons*, with associated connection weights that, after receiving a proper training, are capable of achieving a desired response to new inputs. Its ability of learning from examples makes ANN an extremely powerful programming tool when domain rules are not completely certain or when some amount of inaccuracy or conflicting data exist [13].

2. Artificial neural networks as simulation metamodels

The main purpose of simulation metamodeling is to reduce the cost, time, and amount of effort required during a simulation analysis. A metamodel, or response surface, is an approximation of the input/output function implied by the underlying simulation model. It is usually a supplementary

* Corresponding author. Tel.: +1-205-348-3337; fax: +1-205-348-7162.
E-mail address: dfonseca@coe.eng.ua.edu (D.J. Fonseca).

1474-0346/02/$ - see front matter © 2003 Elsevier Ltd. All rights reserved.
doi:10.1016/S1474-0346(03)00005-3

model that can be alternatively used to interpret a more detailed model.

The almost simultaneously published works of Hornik et al. [6] and Funahashi [4] rigorously demonstrated that multi-layer feed forward neural networks are capable of approximating any measurable function to any desired degree of accuracy. Pierreval [15] used neural networks to model the simulation of manufacturing shops. He proposed a neural network to approach the selection of dispatching rules on a simplified flow shop. The network was trained with simulated data, and, afterwards, it was able to select the best dispatching rules for new cases. In addition, Badiru and Sieger [2] implemented an ANN-based simulation meta-model to predict future cash flow values for cost estimation, using the initial investment and the interest rate as uniformly random variables.

Finally, the recent studies by Kilmer [8] and Kilmer et al. [9,10] exposed the use of supervised neural networks as a metamodeling technique for discrete, stochastic simulation. The first work provides a foundation and methodological guidelines for developing ANN metamodels, while the other papers report the creation of ANN metamodels for simulating a hospital's emergency room, and a factory's inventory warehouse, respectively.

3. Development of the neural network models

3.1. Problem definition and training samples generation

A problem from Askin and Standridge [1] was extracted, enhanced, and a degree of uncertainty added to the processing times for this study. The simulation software package Arena was used to create the set of examples necessary to train and test the prototype networks. Only feed forward, multi-layered, fully connected networks were considered in the study. The models were trained through the back-error-propagation (BEP) learning algorithm. BrainMaker, a commercial ANN software package, was chosen as the development shell for this project.

The simulated job shop consisted of the use of four machines, A, B, C, and D to manufacture four types of parts or orders. Job arrivals were modeled according to an exponential distribution with a mean of 5.5 time units (TU).

Table 1
Processing times per job and machine type (mean, standard deviation, in TU)

Job type	Proportion in arrivals (%)	Machine A	Machine B	Machine C	Machine D
1	40	9,1.3	7,1	8,1.2	–
2	15	7,1	8,1.2	9,1.3	7,1
3	20	–	9,1.3	7,1	9,1.5
4	25	9,1.5	4,1	–	8,1

Table 1 shows the arriving job types proportions, as well as the parameters of the normally distributed processing times.

Simulation runs were executed for 480 TU. Output statistics for 900 different and randomly selected job shop situations were collected from 20 independent replications. The first 800 problems were used for training and testing the ANNs during the design step, while the other 100 scenarios were reserved for validation purposes. The simulation analysis consisted of estimating the average flowtimes when orders follow different machine sequences. All other parameters, even stochastic, were kept unchanged throughout the simulation runs.

3.2. ANN development

ANNs with one hidden layer (as shown in Fig. 1) were pursued in this study, although some tests on two-hidden-layer networks were also performed. Data representation was a critical issue that directly affected the resulting ANN architecture. The number of input neurons required to represent any given job type sequence combination relied on the definition of these sequence codification schemes (SCS). Several codification rules were developed and tested in this study following conventional guidelines given in the available literature [5,11]. A brief description of each one is provided as follows.

The first devised SCS represents any given machine by using a pair of numbers: $+1$ and -1. Thus, machine A is represented by $(-1, -1)$, machine B by $(-1, +1)$, machine C by $(+1, -1)$, and machine D by $(+1, +1)$. Table 2 illustrates how a job shop situation is mapped into this SCS. The application of this procedure to all possible job types and machines required a total of 26 digits.

The second scheme takes into account the type of job and the number of operations required by each job type. A '$+1$' represents an 'active' machine for that operation, while a '-1' indicates that the corresponding machine is 'inactive' or 'not used' for a given operation. For instance, if the first operation for job type 1 is to be performed on machine A, it is then represented by $(1, -1, -1)$. This means that machine A is the workcenter used for that operation, while machines B and C are not. If such an operation is instead

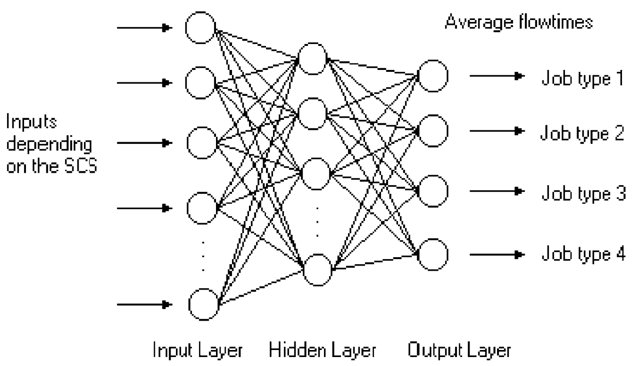

Fig. 1. General ANNs architecture.