Multilayer perceptron for simulation models reduction: application to a sawmill workshop

Philippe Thomas, André Thomas

Accepté à: Engineering Applications of Artificial Intelligence

Multilayer Perceptron for Simulation Models Reduction: Application to a Sawmill Workshop

P. Thomas[1], A. Thomas
Centre de Recherche en Automatique de Nancy (CRAN-UMR 7039),
Nancy-University, CNRS
ENSTIB 27 rue du Merle Blanc, B.P. 1041
88051 Epinal cedex 9 France

Abstract

Simulation is often used to evaluate supply chain or workshop management. This simulation task needs models, which are difficult to construct. The aim of this work is to reduce the complexity of a simulation model design. The proposed approach combines discrete and continuous approaches in order to construct speeder and simpler reduced model. The simulation model focuses on bottlenecks with a discrete approach according to the theory of constraints. The remaining of the workshop must be taken into account in order to describe how the bottlenecks are fed. It is modeled through a continuous approach thanks to a neural network. In particular, we use a multilayer perceptron. The structure of the network is determined by using a pruning procedure. For validation, this approach is applied to the modelisation of a sawmill workshop.

Keywords: multilayer perceptron, reduced model, simulation, neural network, supply chain, ANN

[1] Corresponding author. Tel: +33329296173 / fax: +33383684437. E-mail address: philippe.thomas@cran.uhp-nancy.fr (P. Thomas).

1. INTRODUCTION

Simulation is used in many goals. One of them is to evaluate supply chain or workshop performance. There are three different ways of measuring this performance: analytical models (queuing theory,…), physical experimentation (lab platforms, industrial pilot implementation, …), Monte Carlo methods (simulation or emulation) (Thierry *et al*., 2008). Analytical methods are generally impracticable because the mathematical model corresponding to a realistic case is often too complex to be solved, and physical experiments suffer from technical and cost-related limitations. Simulation is the better approach to model and analyze performance for large-scale cases. In the simulation model, the number of 'objects' of the model and the number of events can be very large. Consequently, the first problem could be the time needed to build the model and the simulation duration on a computer can be unacceptable for operational use. Thus, it is necessary to reduce the model size (Thierry *et al*., 2008).

On one hand, constructing a simulation model is a complex task that can take modelers a lot of time. Effectively, simulation models of actual industrial cases are often very complex and the modelers encounter problems of scale (Page *et al*., 1999). Thus, numerous authors have expressed interest in using simplest (reduced/aggregated) models of simulation (Ward, 1989; Musselman, 1993; Pidd, 1996; Brooks and Tobias, 2000; Chwif *et al*., 2006).

On the other hand, to establish and to initialize 'predictive schedule' or 'reactive schedule', the knowledge of the evolution of resources states, (WIP (work in process), and

queues) are needed. This knowledge can be obtained by using a simulation model. Reduced models can be very useful, because they are quickly parameterized and simulated.

Furthermore, at this level of planning (master production schedule), load/capacity balancing is obtained via the 'management of critical resource capacity' function or 'rough-cut capacity planning' (RCCP), which essentially deals with bottlenecks (Vollmann *et al*., 1992). Goldratt and Cox, in 'The Goal' (1992) put forward the 'theory of constraints' (TOC), which proposes to manage all the workshops by bottlenecks control. Thomas and Charpentier (2005) have shown that a good method to build a simulation model would be to reduce the model according to the TOC.

Moreover, neural networks have been used in all application areas of the manufacturing: scheduling (Akyol and Bayhan, 2007), design of manufacturing process (Cakar and Cil, 2004), …

Therefore, the main goal of this work is to propose a design approach for simulation models, which would be less time consuming and simpler for the modelers, and which could be partially automated. This approach is based on the learning capabilities of neural networks and on the TOC.

The rest of the paper is structured as follows. The second part contains a brief bibliography overview and, in the third part, the proposed approach of the reduction model and multilayer perceptron is presented. The fourth part is devoted to the validation of the proposed approach in an industrial application, which is a sawmill flow shop case.

2. A BIBLIOGRAPHY OVERVIEW

2.1. On supply chain simulations

One main goal of the supply chain simulation is to evaluate the performance of supply chain management in order to support decision making at three levels:

- strategic level (designing or redesigning a supply chain, localization of factories and warehouses, partners selection, etc.),
- tactical level (validation of the global forecasted production capacities according to forecasted demand), and
- operational level (control policies, scheduling, cooperation policies on the shop floor, etc.).

The simulation model must be constructed according to its use and the supply chain to be modeled.

Kleijnen and Smits (2003) distinguish four simulation types for supply chain management:

- spreadsheet simulation (may be part of production control software),
- system dynamic (may explain the bullwhip effect),
- discrete-event dynamic systems (DEDS) simulation (may predict fill rate values), and
- business game (may educate and train users).

Spreadsheets have been used to implement manufacturing resource planning (MRP), but this type of simulation is often too simple and unrealistic (Kleijnen, 2005).

System dynamic is based on the work of Forrester (1961). In this approach, companies are seen as systems with six types of flows (materials, goods, personnel, money, orders, and

information) and different stocks. Managerial control is realized through the changing of rate variables. The feedback principle plays a crucial role in this approach (Kleijnen, 2005).

A DEDS simulation is more detailed than the preceding ones. DEDS concerns the modeling of a system by a representation in which the state variables change instantaneously according to event occurring. Moreover, it takes into account uncertainties (Law and Kelton, 2000).

A business game is a simulated world that may represent a supply chain and its environment. It is used for educational and research goals (Kleijnen, 2005).

The two main difficulties encountered during the design step of a supply chain simulation model are related to the size of the system and the complexity of the control system. A supply chain is composed of a group of enterprises, composed in turn of a group of factories, composed of a group of workshops, etc. Moreover, modeling the behavior of the leading policies of each enterprise and the relationships between them is needed (Thierry *et al.*, 2008). This fact implies that the duration of one simulation may become unacceptably long to be usable. The same difficulty has been highlighted by Thomas and Charpentier (2005) concerning workshop. Therefore, it may be useful to reduce the size of the model. Different ways can be used to perform the model reduction:

- abstraction, which allows the complexity of the model to be reduced and preserves the validity of the results (Frantz, 1995),
- aggregation, which is a form of abstraction where a group of data or variables with common characteristics can be replaced by aggregated data or variables (Aladanondo and Mercé, 1991), and

- reduction of the number of events, where a part of DEDS is replaced by a variable or a formula (Zeigler, 1976).

2.2. On model reduction

Innis and Rexstad (1983) have listed 15 simplification techniques for general modeling. Their approach is composed of four steps: hypotheses (identify the important parts of the system), formulation (specify the model), coding (build the model), and experiments. Based on these works, different approaches have been proposed.

Brooks and Tobias (2000) suggest a 'simplification of models' approach for cases where the indicators to be followed are the average throughput rates. They suggest an eight-stage procedure. The reduced model can be very simple and then an analytical solution becomes feasible and the dynamic simulation redundant. Their work is interesting, but is valid in cases where the required results are averaged and where the aim is to measure throughput. It is not interesting to follow the various events taking place in the work center (WC).

Leachman (1986) has proposed a model for use in the semiconductor industry, which uses cycle time as an indicator. This model has been improved by Hung and Leachman (1999). They propose a technique for model reduction to be applied in large wafer fabrication facilities. They use 'total cycle time' and 'equipment utilization' as decision-making indicators to do away with the WC. In their case, these WCs have a low utilization rate and a fixed service level (they use the standard deviation of batch waiting time as a decision-making criterion).

Tseng *et al*. (1999) compare the regression techniques applied to an 'aggregate model' (macro) by using the 'flow time' indicator. They suggest reducing the model by mixing the

'macro' and 'micro' approaches, so as to minimize errors in complex models. Here again, for the 'macro' view, they deal only with the estimation of flow time as a whole. For the 'micro' approach, they construct an individual regression model for each stage of the operation to estimate its individual flow time. The cumulative order of flow time estimates is then the sum of the individual flow times. They, then, try to mix the macro and micro approaches. These different approaches simplify the model by using a macroscopic view of the system and by optimizing a macroscopic indicator (total cycle time, flow time…)

Li *et al*. (2009) propose a reduction model approach based on the aggregation of machines on the production line. They build a complete model of the production line and, if the last two machines correspond to a serial line, they aggregate them. The same is performed with the first two machines if they correspond to a serial line. These aggregation steps may be performed recursively and they are denoted backward and forward aggregation, respectively. If the two machines to be aggregated follow a Bernoulli model or an exponential model, an analytical investigation allows the production rate of the new aggregated machine to be determined. If not, a simulation phase must be performed to determine an empirical formula for the production rate.

Some works (Doumeingts *et al*., 1987; Hwang *et al.*, 1999) use Petri nets as tool in order to simplify network structures by using macro-places which represent complex activities associated with function groups.

To simplify models, some works have studied the use of a continuous flow model based on gradient estimation for stochastic systems in order to approximate discrete manufacturing environments (Ho, 1987; Suri and Fu, 1994). Other authors use metamodels (linear regression, splines, Kriging, etc.) to perform a simulation model (Kleijnen and

Sargent, 2000). Neural networks can be viewed as a type of metamodel (Barton, 1994; Pierreval, 1996; Kleijnen and Sargent, 2000). In addition, neural networks have proved their abilities to extract models from experimental data (Thomas *et al*., 1999). Therefore, the use of neural networks has emerged recently as an interesting approach within the framework of the supply chain or workshop management (Shervais *et al*., 2003; Chiu and Lin, 2004).

2.3. On neural network in manufacturing

Neural network approaches have been used in all application areas of the manufacturing. Zhang and Huang (1995) have noticed that neural networks are used in monitoring and diagnosis, process modeling and control, group technology, engineering design, quality assurance, robotics, scheduling, or process planning areas.

Different typologies of neural network have been used for dealing with scheduling problem (Akyol and Bayhan, 2007). Hopfield network and its extensions are used to solve optimization problems. So, many works use Hopfield network to determine static scheduling by minimizing the sum of all the starting times of each job's last operation (Foo and Takefuji, 1988; Foo *et al*., 1995), by minimizing makespan (Willems and Brandts, 1995), or by minimizing the weighted sum of the earliness and tardiness penalties (Akyol and Bayham, 2005). These approaches are generally infeasible for large size problems and may generate constraints-violating solutions. Competitive network and self organizing map have also be used to deals with the same problem. Fang and Li (1990) use competitive networks in order to minimize total tardiness. The use of competitive network needs the definition of equations of motion for the problem constraints and an energy function that

converges to stable state. Chen and Wang (2009) develop a self organizing map–back propagation network to estimate the remaining cycle time of every job in a semi conductor manufacturing factory. Multilayer perceptrons are also used in scheduling applications in order to select a suitable scheduling strategy (Geneste and Grabot, 1997), or designing a scheduling software (Feng *et al*., 2003).

Some works focus on one aspect of scheduling problems. Lin and Hwang (1999) study dynamic task allocation and use two multilayer perceptrons to allocate the task between human and computer. Dispatching rules selection in a job shop has been investigated by El Bouri and Shah (2006) which minimize the makespan and the mean flow time with two different neural networks. Kuo *et al*. (2007) address the same problem by focusing on the construction of the learning data set. Mouelhi-Chibani and Pierreval (2010) determine the parameters through simulation optimization to perform the dispatching rules selection.

The design of manufacturing process is another important area of neural network applications. Cakar and Cil (2004) determine the number of machines in a work center in function of priority rules. The inputs of the multilayer perceptron used are machine utilization rate, percentage of the late parts, and mean values of flow time, tardiness, and completion time. Vosniakos *et al*. (2006) associate multilayer percetron and genetic algorithm to analysis and design manufacturing cells. Araz *et al*. (2008) focus on the determination of the optimum kanban parameters by using a multilayer perceptron for generating simulation metamodels.

This area is related to the process metamodeling which is the core of many works. Chambers and Mount-Campbell (2002) propose to model each component of a process by a neural model and to associate them in order to optimize the complete process.

In order to estimate important parameters, some authors propose to use neural networks as metamodels. Fonseca and Navarese (2002) determine the manufacturing lead times for orders simultaneously processed in a job shop. The time-throughput is estimated by using multilayer perceptron for single/multi product manufacturing environments (Yang, 2010). Kutschenreiter-Praszkiewicz (2008) uses Radial Basis Function to estimate time consumption in machining.

In all these applications, the choice of the structure of the neural network is always a complex task and determining a suitable or near-optimal structure for a neural network has been called a “black art” (Branke, 1995). However, some authors try to respond to this question. Sukthomya and Tannock (2005) investigate the selection of inputs, and they attempt to provide guidelines for the training of neural networks to model complex industrial processes. Khosravi *et al*. (2010) try to construct neural metamodels with optimal structure. Moreover, they build prediction intervals for point predictions of neural metamodels.

For multilayer perceptron, the determination of the near-optimal structure is a well-known problem investigated by many authors. Two main approaches can be used: Constructive approaches (Chentouf and Jutten, 1996; Rivals and Personnaz, 2003) and pruning approaches (Hassibi and Stork, 1993; Drucker, 2002).

Akyol and Bayhan (2007) recall the main advantages and disadvantages of the Hopfield networks, competitive networks and multilayer perceptrons.

The main advantage of Hopfield networks is their massive parallelism architecture when their main disadvantages are that they may converge to local optimum, the ways of

incorporating constraints into the energy function and the termination criteria affect the quality of the results, and the translation of the problem into the energy function is difficult.

Competitive networks are best applicable to optimization and classification problem and by using competitive learning rule, the penalty terms are handled explicitly therefore the energy function is simplified and the time required in obtaining coefficient is reduced. However, equations of motions need to be derived before solving the problem, competitive networks cannot be applied to simplify the energy function of all scheduling problems and their convergence should be analyzed carefully.

Multilayer perceptrons are universal approximators, which have better generalization capabilities to capture complex relationship between inputs and outputs. However, their main drawbacks are that gradient-based training techniques may be trapped into local minima, the generation of training set is time consuming, and overlearning degrades the performance of the network. The problem of local optimum may be attenuated by using an adapted initialization algorithm and the determination of the optimal structure allows avoiding the overlearning problem.

3. MODEL REDUCTION PROCESS

3.1. Proposed approach

The proposed approach is based on the association of discrete event models and continuous models (neural network) in order to design a simulation model. Our objective is to maximize the bottleneck utilization rate and, at the same time, simplify simulation model construction for modelers.

The reduction algorithm proposed is an extension of those presented by Thomas and Charpentier (2005). The main goal of this algorithm is to reduce the number of simulation blocks. It is based on the 'theory of constraints' which uses the concept of bottleneck. Many definitions of bottleneck are available in the literature and still more are used in practice. Most of them view the bottleneck as the worst machine, e.g., the machine with the smallest efficiency. Other authors consider the machine with the largest effect on the throughput as bottleneck (Li *et al*., 2009). Here, two particular types of work centers (WC) are defined:

- 'conjunctural bottleneck' (current bottleneck) is a WC that is saturated for the master production schedule (MPS) in the predictive scheduling in question. This means that it uses all of its available capacity,
- 'structural bottleneck' means WC that has often been or is in such a condition. These 'structural bottlenecks' are determined by experience feedback.

The proposed algorithm is presented figure 1 and its main steps are recalled and explained below:

1. Identify the WC which is the structural bottleneck. As said before, this one has been the main capacity constraint for several years (according to the experience of production manager).
2. Identify the conjunctural bottleneck for the bundle of MOs of the MPS under consideration.
3. Among the WCs not listed in 1 and 2, identify the one (synchronization WC) that satisfies the following two conditions:
    - necessary at least for one of the MOs using a bottleneck, and

- widely used considering the whole body of MOs

4. If all MOs have been considered, go to 5; if not, go to 3.
5. Model all the WCs that have not been found during the previous steps by using a continuous model, a neural network.

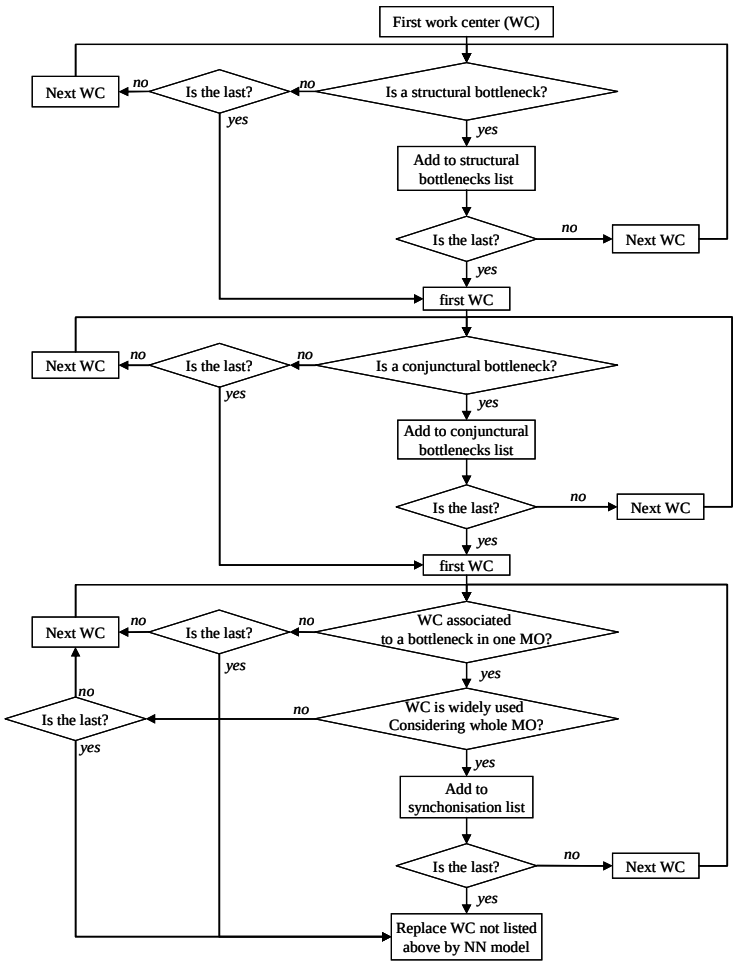

Figure 1. Algorithm used

Hence, the WCs remaining in the model are either conjunctural or structural bottlenecks, or are WCs that are vital to the synchronization of the MOs. All other WCs are incorporated in 'aggregated blocks' upstream or downstream of the bottlenecks. These

'aggregated blocks' are modeled by neural networks which estimate the throughput times between two bottlenecks. These models permit to simulate the alimentation of the bottlenecks and so, to control the bottlenecks.

The main benefits of this algorithm are:

- modelers can focus on the description of the bottlenecks,
- noncrucial parts of the system are modeled with a learning approach (automatization of this modeling step),
- the resulting model is less complex than a complete one, and
- simulation time is shorter than with a complete model.

This paper focuses on step 5 of the reduction algorithm. The bottlenecks are considered here as known.

3.2. The multilayer perceptron (MLP)

The works of Cybenko (1989) and Funahashi (1989) have proved that a multilayer neural network with only one hidden layer using a sigmoidal activation function and an output layer using a linear activation function can approximate all nonlinear functions with the desired accuracy. This result explains the great interest of this type of neural network, which is called 'multilayer perceptron.' In this work, the objective is to model the throughput times of parts between two bottlenecks by using information given by the system. It is assumed that this throughput time could be approximate with a nonlinear function obtained with a MLP.

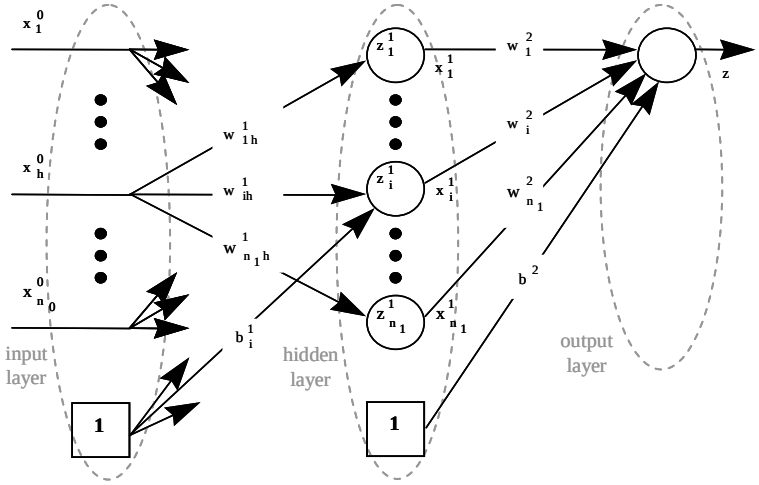

Figure 2. Structure of the multilayer perceptron

The structure of the multilayer perceptron is recalled here. Its structure is shown in figure 2. The neurons of the first (or input) layer distribute just the n_0 inputs $\{x_1^0, \cdots, x_{n_0}^0\}$ of the MLP to the neurons of the next (hidden) layer. A special input neuron (depicted by a square in figure 2) represents a constant input equal to 1, and it is used to represent the biases or thresholds of the hidden layer. The output of the neurons of the hidden and of the output layers is given by a so-called 'activation function' of the weighted sum of its inputs. The activation function of the hidden neurons is the hyperbolic tangent when the activation function of the output neuron is a linear one. The form of this neural network is given, for single output, by:

$$z = \sum_{i=1}^{n_1} w_i^2 . x_i^1 + b = \sum_{i=1}^{n_1} w_i^2 . g\left(z_i^1\right) + b = \sum_{i=1}^{n_1} w_i^2 . g\left(\sum_{h=1}^{n_0} w_{ih}^1 . x_h^0 + b_i^1\right) + b \qquad (1)$$

where x_h^0, h = 1, ..., n_0, are the inputs of the network, w_{ih}^1 and b_i^1, i = 1, ..., n_1, $h = 1, \cdots, n_0$, are the weights and biases of the hidden layer, x_i^1, i = 1, ..., n_1 are the outputs of the hidden neurons, w_i^2 and b are the weights and bias of the output neuron.

Kleijnen and Sargent (2000) have proposed a modeling process that can be subdivided into 10 steps:

- determine the goal of the model,
- identify the inputs and their characteristics,
- specify the domain of applicability,
- identify the output variable and its characteristics,
- specify the accuracy required of the model,
- specify the model's validity measures and their required values,
- specify the model and review this specification,
- specify a design,
- fit the model, and
- determine the validity of the model.

In this work, these different steps are used to construct the neural network. The four first steps are related to the design of the input and output layers. The output neurons represent the information to model when the input neurons correspond to the data available in order to model the considered system.

The three last steps are related to the design of the hidden layer and to the learning of the parameters. The determination of the hidden layer and the learning of the parameters are

performed simultaneously. For this, the learning starts from an overparameterized structure and it is performed in three steps:

- initialization of the weights and biases of the oversized structure,
- learning of the parameters,
- pruning of the spurious parameters.

The initialization of the weights and biases is performed by using an evolution of the Nguyen-Widrow (1990) algorithm proposed by Thomas and Bloch (1997). This algorithm permits to associate a random initialization of weights and biases to an optimal placement in the input and output spaces. This method is similar to the slice linearization and permits to avoid the initial saturation of hidden neurons.

The learning algorithm used is the Levenberg-Marquard algorithm with a robust criterion (Thomas and Bloch, 1996). The Levenberg-Marquard algorithm permits to associate the speed of the Hessian methods to the stability of the gradient methods. This is performed by adding a parameter multiplied by the identity matrix in order to permits the inversion of the Hessian matrix even if it is singular. The tuning of this parameter during the learning permits to the Levenberg-Marquard algorithm to work as a gradient descent algorithm when this parameter is great and as a Gauss-Newton algorithm when this parameter is small. The use of a robust criterion permits to avoid the influence of outliers and, has a regularization effect in order to prevent overfitting.

The pruning algorithm used is the Neural Network Pruning for Function Approximation (N2PFA) algorithm (Setiono and Leow, 2000). This algorithm uses the mean absolute deviation (MAD) to measure the performance of the neural network. It is performed into two main steps. In the first one, the spurious hidden neurons are pruned, and in the second

one, the feature selection is performed. The strategy for eliminating an hidden neuron (first step) or an input (second step) is the same and is very simple and fast. During the first step, the hidden neuron i (i = 1, ..., n_1) is deleted (by vanish the weight w_i^2) and the resulting structure is evaluated by calculating the MAD values for the learning and validation data sets. The best resulting structure is compared with the initial one, and, if its MAD values are not so degraded, the considering hidden neurons is remove and the procedure is repeated until no hidden neurons can be removed. Else, the initial structure is keep.

The same work is performed in a second step on the input neurons.

4. ILLUSTRATION OF THE PROPOSITION

For illustration, we use the proposed approach to build a simulation model of a sawmill workshop. The main objective of sawmill is to cut tree trunks into planks of different sizes. In this actual case, managers need a tool to help them in their weekly decision-making Master Production schedule (MPS) process. This work is resulting from collaboration with the sawmill which want:

- to evaluate the effectiveness of its MPS,
- to maximize its load rate, and so, its global productivity,
- to explain some unexplained congestion phenomena of the trimmer WC.

A first work (Thomas and Charpentier, 2005) with a complete model has permit to represent the congestion phenomena and to use this representation in order to improve the load rate. This model, which is recalled in section 4.2a, has permit to show that a load rate of the bottleneck too high (higher than 60%) degrades the productivity of this bottleneck,

and so, the productivity of the sawmill. The difficulty is that the bottleneck is the last work center of the sawmill but all the influent factors on the productivity of the bottleneck depends of the first work center.

However, this complete model is unusable on a real case for the dynamic evaluation of the MPS because of the time needed to modified it.

4.1. Overview of the sawmill

At the time of the study, the sawmill has a capacity of 270,000 m^3/year, a turnover of €52 million and 300 employees.

This workshop can be described from a process point of view. This sawmill can be represented by two linear parallel flows for the main and secondary products. This fact, associated to the variation of log dimensions lead the process to be non-linear. Therefore, the physical industrial production system can be divided into three main parts. To understand the functioning of the process, the course of a log from its admission into the process to its exit in planks form will be described.

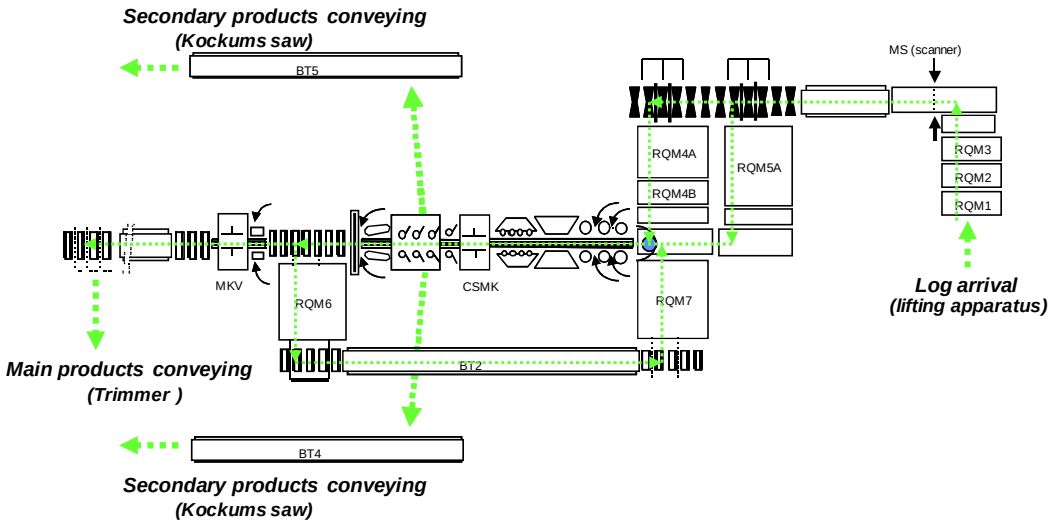

Figure 3. The Canter line

The first part of the process corresponds to the Canter line, which is presented figure 3. Dashed arrows indicate the products flow. The considered log is taken into the process by using conveyors RQM1, RQM2, and RQM3. According to its characteristics (determined by scanner MS), the log is driven to conveyors RQM4 or RQM5, which are used as input inventory for the Canter line. Only RQM5 is used here in order to simplify the presentation. After that, logs go on the first canter's machine and later on the CSMK saw, which transforms logs into square-shaped parallelepipeds (figure 4).

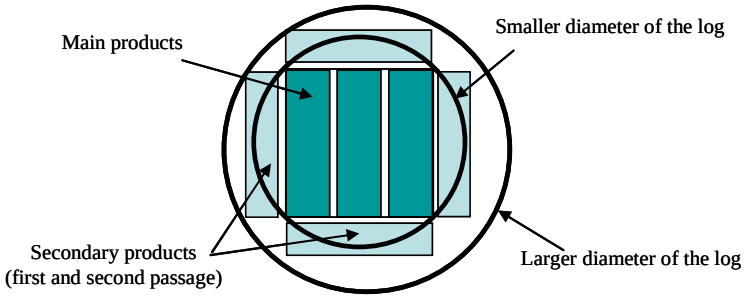

Figure 4. The cutting plan

This first step, which gives the two first sides of the parallelepipeds, produces two planks (called secondary products), which are taken out of the Canter line by the BT4 and BT5 conveyors. The log is then driven on the RQM6 conveyors, then rotated 90°, and stored in RQM7 awaiting its second passage on the CSMK saw. After the second passage, the squared is completed, and two other secondary products are taken out of the Canter line by the BT4 and BT5 conveyors toward the second part of the process, the Kockums line. The squared log is cut on the MKV saw into three planks (called main products). These

main products are driven to the third part of the process, the trimmer line. The cutting of the log into main and secondary products is described in the cutting plan (figure 4).

Figure 5 shows the second part of the process, where the main machine is the Kockums saw. Only secondary products are driven on this part. The secondary products are taken in the line by the BT4 and BT5 conveyors. They are cut by the QM11 saw, after which they reach the Kockums saw, which optimizes the planks according to the products needed. The alignment table is used as the input inventory of the Kockums saw. The secondary products are finally sent to the third part of the process by the exit conveyor.

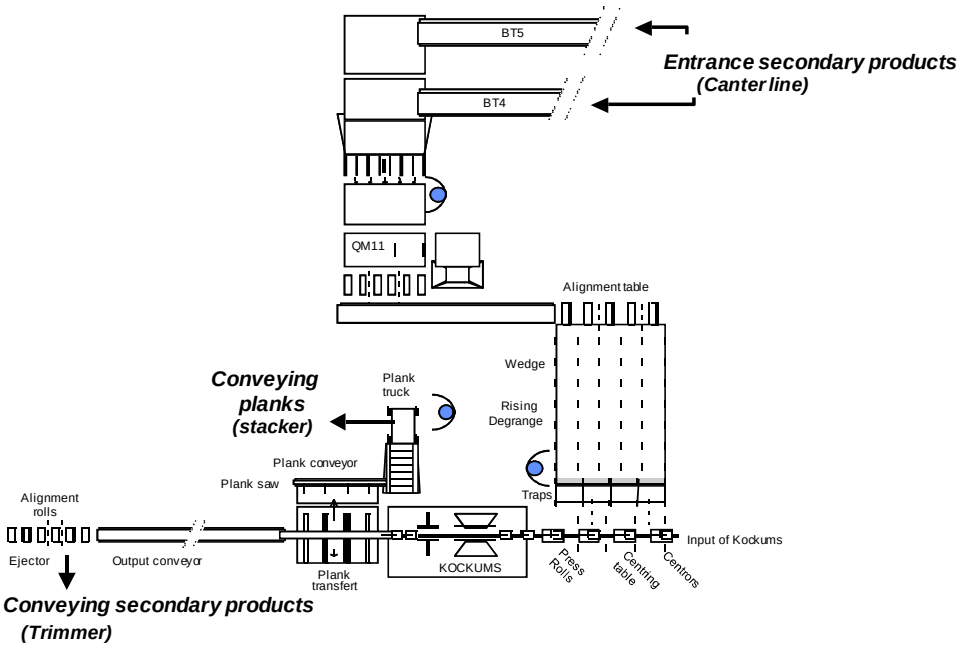

Figure 5. The Kockums line

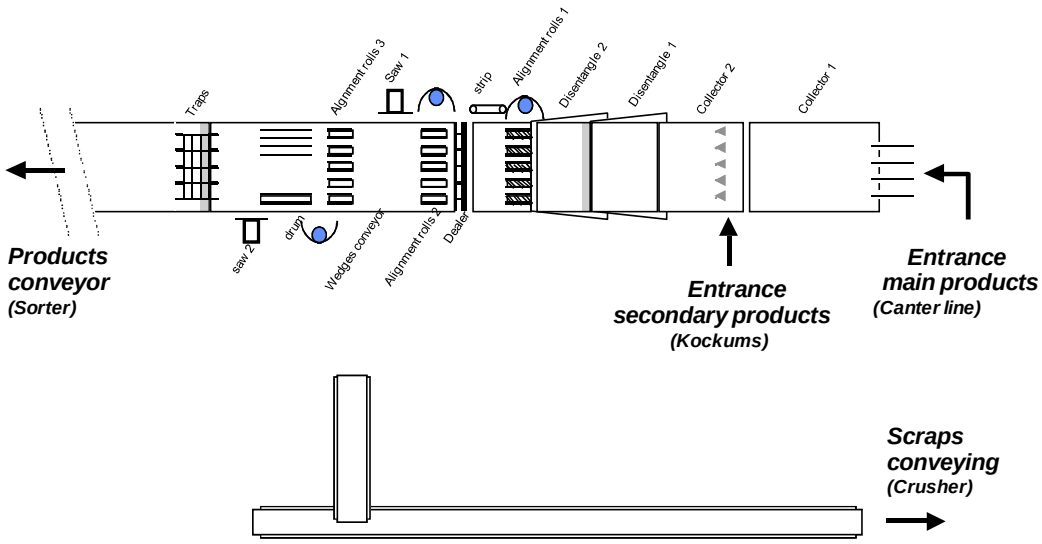

Figure 6. The trimmer line

The third part of the process is the trimmer line, which is presented in figure 6. This line performs the final operation of cross cutting. This operation consists in cutting up products to length. The inputs of the line are from collectors 1 and 2, which collect the secondary and main products from Kockums and Canter lines respectively. Saw 1 is used to perform default bleeding and Saw 2 cuts up products to length.

A previous work (Thomas and Charpentier, 2005) has shown that this last machine, the trimmer saw, is the bottleneck of the entire process, and, as said previously, the productivity of the trimmer depends to the decisions taken on the canter work center. So, the impact of bad decisions are seen too much late to be corrected. So in order to evaluate decisions, managers need a simulation tool.

4.2. The simulation models

a. The complete model

The complete model of the sawmill process (figure 7) is constructed with the Arena® software and consists of different modules. The first module is used to model the log arrival, which follows a homogeneous Poisson process with a mean of 20. In this module, the characteristics of the log, which are measured by the scanner (figure 3), are associated with it. In the simulation model, 2000 logs are presented at the entry of the process. The dimensions of these logs follow uniform distributions. At starting time, the process is empty.

A second module, the 'input sorter', directs the logs to either RQM4 or RQM5, according to their characteristics. It may also eject the log out of the process if it is machine-gunned or if its dimensions are out of range. The logs go to the next module, which models the RQM4 and RQM5 queues. Conveyors RQM4, RQM5, and even RQM7, are used as input inventory for the Canter line. Two other modules are used for the simulation of the Canter line and the passage of the squared log in RQM7. The Canter line model uses two submodels for the management of main and secondary products. The Canter line has three outputs, which lead to the Kockums line for the secondary products and to the trimmer line for the main products.

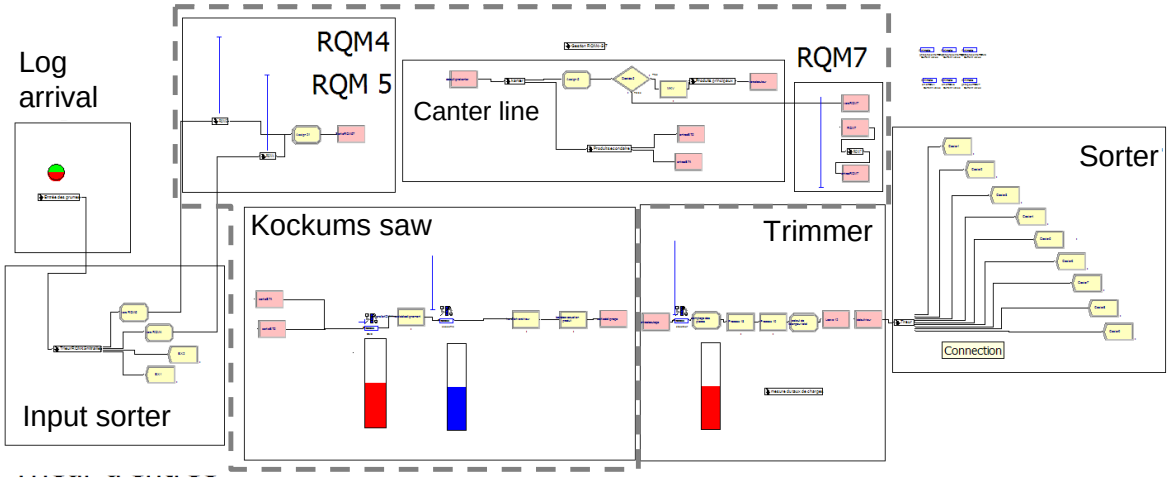

Figure 7. The complete model

The other modules, which correspond to the core of the process, are the simplest. They are used to model the Kockums and the trimmer lines and the last module is used to model the sorter of products into different racks.

The different submodels make the model more complex that it appears in figure 7. In particular, construction of the submodel used to manage the priority rules for choosing the input inventory that supplies the Canter line is a very complex task. The simulation ended when all the logs (with the exception of ejected ones) are cut into planks. The simulation results of this model will be used in section 4.3.

This model has permit to explain some phenomena of congestion of the trimmer and so, to improve the productivity of the sawmill. However, it is not useful for the dynamical evaluation of the MPS. In fact, the construction of the complete model needs one day for an experimented people. But, the time of readjustment of the initial model each time an event forces to re-use the model is prohibitive. An event may force to change attributes, distribution laws, times, and even, may need to modify some flow or work centers… And,

this work must be performed in a very short time because we need to react quickly to the event. that's why a complete model is not pertinent and the use of a reduced model which is quicker to construct is needed.

b. The reduced model

As said previously, the bottleneck of this line is the trimmer. Consequently, modeling the function of the inventories RQM4, RQM5, and RQM7, and of the Canter and Kockums lines is unnecessary. Furthermore, the part surrounded by the gray dashed line in figure 7 gave no direct and useful information for the evaluation of the MPS. In fact, only the arrival times of the products in the trimmer queue are useful for simulating the load of this bottleneck, and this is the reason for using a multilayer perceptron.

According to the modeling process recalled in chapter 3.2, the specific sawmill neural model could be constructed. To build a neural network, we need to identify the input variables. Thomas *et al.* (2008) collected the available input data which can be classified into three categories: Data related to the products (here the logs), data related to the process and data related to the bill of material or routing (here the cutting plan).

The data related to the products are mainly dimensional ones as length (lg) and three values for timber diameters (diaPB, diaGB, and diaMOY). The thickness of the finished product, may also be used. However, in a previous work (Thomas *et al.*, 2008), it was shown that thickness have no impact on the result and so it is not taken into account in this work.

The data related to the process are the process variables collected at the time of log arrival. In particular, we require the input stock and the utilization rate of the bottleneck,

here the trimmer (Q_trim, and U_trim, respectively). The number of logs present in the process between the inputs of RQM5 and the exit of the Canter line (Q_RQM) is needed. The data related to the routing correspond here to the information related to the cutting plan of the logs which must be cut into main and secondary products. Here the cutting plan (figure 4) divides the log into seven products:

- two secondary products resulting from the first step of the cutting process on the saw CSMK of the Canter line,
- two secondary products resulting from the second step cutting process on the saw CSMK of the Canter line after staying in the RQM7 queue, and
- three main products resulting from the third step of cutting process on the saw MKV of the Canter line.

Saws CSMK and MKV belong to the Canter line. These seven products can be classified into three categories, according to the location (CSMK or MKV) and the time during the cutting process (first or second cutting). This information is given by the two variables (prod and Step). The “prod” variable indicates whether products are main or secondary ones. The “Step” variable indicates whether the secondary products are performed during the first or second step (before or after the logs went along the RQM7 queue).

Consequently, the neural networks input variables are: Lg, diaGB, diaMOY, diaPB, Prod, Step, Q_trim, U_trim, and Q_RQM. In our application, 12775 products are simulated with the complete model. These data are used to fit the behavior of the reduced model to the complete one.

The next step is to identify the output variable. Our objective is to estimate the delay (ΔT) corresponding to the duration of the throughput time for the 12775 products. ΔT is

measured between the process input time and the trimmer queue input time. Hence, ΔT is the output variable of the neural network:

$$\Delta T = \sum_{i=1}^{15} w_i^2 . g\left(\sum_{h=1}^{9} w_{ih}^1 . x_h^0 + b_i^1 \right) + b \quad (2)$$

To specify the model, the number of hidden neurons needs to be determined. Therefore, a weight elimination method, N2PFA, is used to remove spurious parameters (Setiono and Leow, 2000). As explained in the part 3.2, the N2PFA algorithm uses the mean absolute deviation (MAD) to determine the effectiveness of the network. This algorithm works in two steps. In the first step, it prunes the spurious hidden neurons. During the second step, the spurious inputs are pruned. In order to avoid an early stopping of the algorithm which drives to an overparametrized structure, a parameter must be tuned which permits a slightly degradation of the MAD values in exchange of the suppression of one neuron. This parameter is tuned to 0.025. With this choice, the deletion of one neuron which degrades the MAD values of 2.5% is accepted.

Therefore, the learning begins with a structure using 15 hidden neurons (2), which correspond to 166 parameters.

The learning of the network is supervised. Hence, it is necessary to divide the database into two datasets, namely, learning and validation. The database is constructed with the complete model, which is used as reference. To fit the model, we use the learning algorithm called the 'Levenberg–Marquard algorithm with robust criterion' (Thomas and Bloch, 1996). The learning approach corresponds to a local search of a minimum and the results may differ according to the initial weights. To evaluate the dispersion of the results, 50 different sets of initials weights are used. The pruning procedure led to the preservation of

the nine inputs in 86% of the cases. In the other cases, only the input U_trim (utilization rate of the trimmer) is removed. The number of hidden neurons after pruning varied from 4 to 14. Figure 8 presents the distribution of the hidden neurons number during the 50 trials.

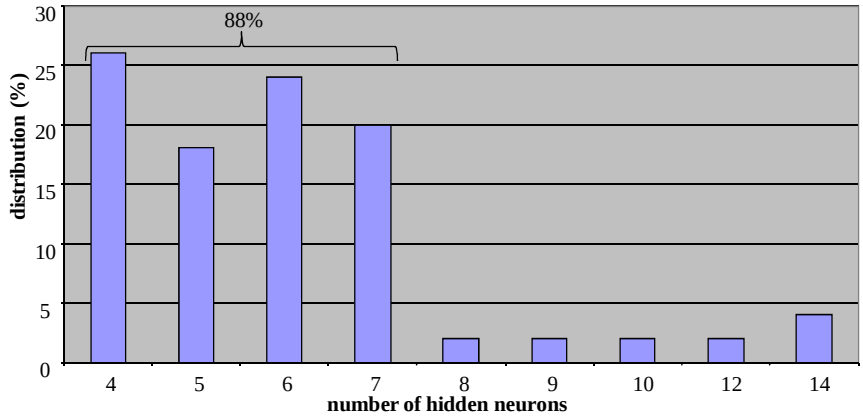

Figure 8. Distribution of the hidden neurons number

This figure shows that, in 88% of the cases, the number of hidden neurons remaining after pruning ranged from four to seven. Table 1 presents the means and the standard deviations of the residuals for the 50 trials on the learning and the validation data sets.

	Learning residual		Validation residual	
	Mean	StD	Mean	StD
Mean(abs)	0.324	54.230	0.946	55.013
StD	1.133	1.853	1.418	1.784
Min	-4.956	52.555	-4.477	53.826
Max	3.570	62.912	4.704	63.121

Table 1. Means and standard deviations of the residuals

Table 1 shows that, even though the network structure may vary, the results obtained on the learning and validation data sets are very close together. Moreover, when the obtained

residuals in these different cases are studied, we notice that the worst results are generally obtained when few hidden neurons are pruned. However, the best results are obtained when all the nine inputs are preserved and only four to seven hidden neurons are retained. The selected structure used nine input neurons and seven hidden neurons, corresponding to 78 parameters. With this structure, the means of the obtained residuals on the learning and the validation data sets are very close to 0 (0.0012 and 0.0865, respectively). The standard deviations of the residuals are 53.075 and 53.826 on the learning and validation data sets, respectively.

To determine if some dynamics existing in the data are not taken into account by the learning process (i.e. the learning process has failed), the correlation between the different inputs and the residuals is performed on the learning and validation data sets (table 2).

		Lg	diaGB	diaMOY	diaPB	Prod	Step	Q_trim	U_trim	Q_RQM
Learning	Mean	0.00704	0.005026	0.00668	0.009879	0.016519	0.004216	0.006994	0.008566	0.010548
	Std	0.01158	0.010112	0.016762	0.026024	0.064165	0.009788	0.009788	0.010978	0.02379
	Min	4.66E-05	4.66E-05	4.75E-05	0.00028	1.12E-05	6.16E-06	0.000121	0	0.000171
	Max	0.05935	0.051441	0.085824	0.13427	0.32708	0.045058	0.044141	0.035216	0.12228
Validation	Mean	0.016825	0.015024	0.015773	0.021277	0.025555	0.015553	0.020086	0.01541	0.020915
	Std	0.010131	0.012723	0.017408	0.028484	0.062529	0.011325	0.015828	0.012356	0.024371
	Min	0.000899	2.93E-05	0.000369	2.94E-07	4.85E-05	0.000716	0.000621	0	0.000789
	Max	0.04093	0.049737	0.087061	0.1446	0.32552	0.050812	0.069208	0.051577	0.1214

Table 2. Correlation between inputs and residuals

Table 2 presents the mean, the standard deviation, the minimum and the maximum values of the correlation coefficient absolute value between the nine inputs and the residuals obtained with the 50 different trials (50 different neural models) (for the learning and validation data sets). These results show the variables “lg” (length), “diaGB (great diameter of the log), “diaMOY” (medium diameter of the log), “Step” (time of production

of the secondary products), “Q_trim” (input queue of the trimmer) and “U_trim” (utilization rate of the trimmer) are always not correlated with the residuals.

The variables “Prod” (main or secondary products) and eventually “diaPB” (smallest diameter of the log) and “Q_RQM” (number of logs present in the process between the input of RQM5 and the exit of the Canter line) are also generally not correlated with the residuals. In some rare cases, these variables and the residuals are correlated. However, these cases correspond to the network structures where too many hidden neurons are keep (more than 7 hidden neurons). It can be noticed that the minimal value for the correlation between “U_trim” (utilization rate of the trimmer) and residuals is 0. This could be attributed to the pruning of the input in some cases. For the selected structure, the coefficient of correlation between inputs and residuals is never more than 0.022505.

Now, we consider the residuals obtained with the selected structure. For the learning data set, the mean of the residual is very close to 0 (0.0012) and it represents an error lesser than 2.7% of the throughput time ΔT. This result shows that the structure of the neural network used is sufficient for learning ΔT. Those obtained on the validation data set corroborate these results. On the validation data set, the mean of the residuals is also close to 0 (0.0865) and the residual represents an error lesser than 2.3% of ΔT. Therefore, we conclude that no overfitting problem occurs and the neural network can estimate the delay for datasets other than the learning one.

Based on these results, we can conclude that our neural network is a good representation of this part of the process.

The neural network model obtained is included into the reduced model shown in figure 9. The modules ‘log arrival,’ ‘input sorter’, and ‘trimmer’ in the reduced model are

identical to those used in the complete one. Only the part surrounded by a gray dashed line is replaced. A comparison between the complete (figure 7) and reduced (figure 9) models shows that the model complexity is greatly reduced. In particular, the different submodels are removed from the reduced model. The reduced model can be so constructed in one hour with an automated procedure.

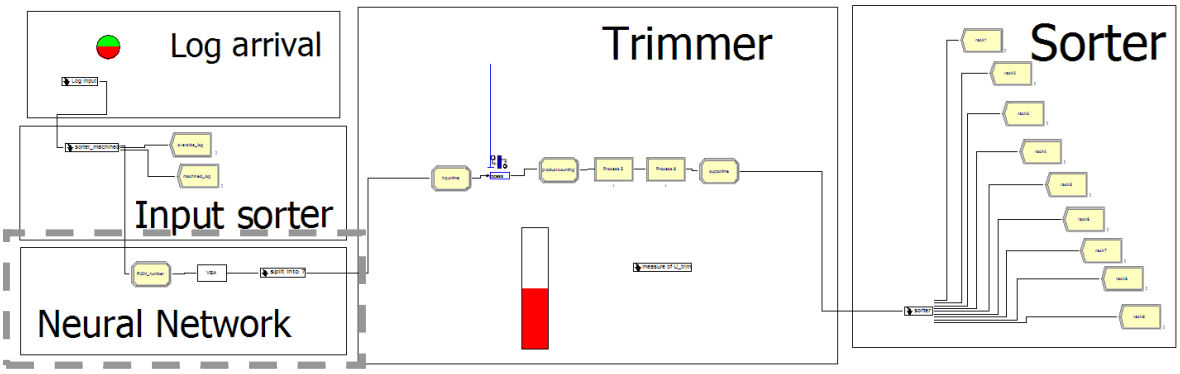

Figure 9. The reduced model

4.3. Evaluation of the reduced model

In this section, we compare the results obtained with the reduced and the complete models.

Figure 10a shows the evolution of the input inventory of the trimmer as a function of time (s). This comparison is performed with two different data sets obtained under the same conditions. Figure 10a shows that the two models presente the same type of queue evolution. However, one difference can be noticed: the trimmer queue value of the complete model between 0 and 500 s increases to 90 before decreasing to values very similar to those obtained with the reduced model. The observed difference between the two

values curves are due to the models initialization. The log arrival in the two models follows a homogeneous Poisson distribution with a mean of 20 s. In addition, many parameters of the models follow a stochastic process. The initializations of the models produce an edge effect, which could explain the differences in behavior between the two models (Thierry *et al.*, 2008; El Haouzi *et al.*, 2008).

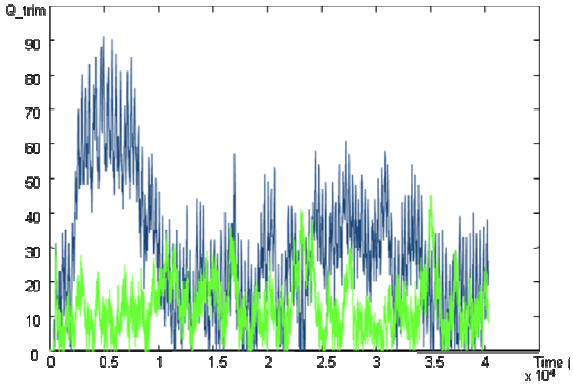

Figure 10a. Input queue of the trimmer

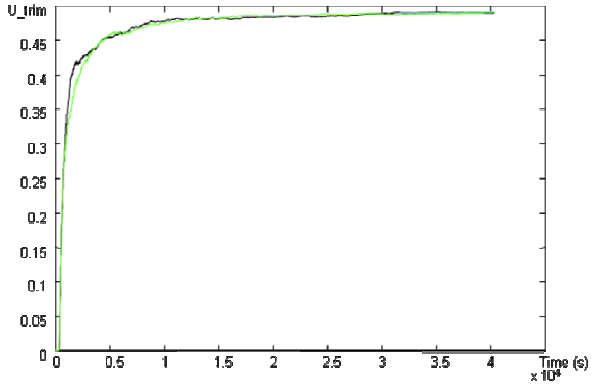

Figure 10b. Utilization rate of the trimmer

(black: complete model; gray: reduced model)

Figure 10b shows the utilization rate of the trimmer as a function of time (s) for the two models (complete in black and reduced in gray). The two models present a similar evolution of the utilization rate and converge to the same value after 1000s.

To confirm that the neural network included in the reduced model has correctly learned the process, we investigate the results obtained by using a homogeneous Poisson process with different means as log arrival rule. As an example, Figure 11 presents the utilization rate of the trimmer when the mean of the Poisson process is 30 s (compared with the 20 s used previously). Figure 11 shows that the two models gave similar results and that the

utilization rates converge to the same value of 33% for the two models (compared with the value of 49% obtained previously). Similar results are obtained when other Poisson processes are used as log arrival rule.

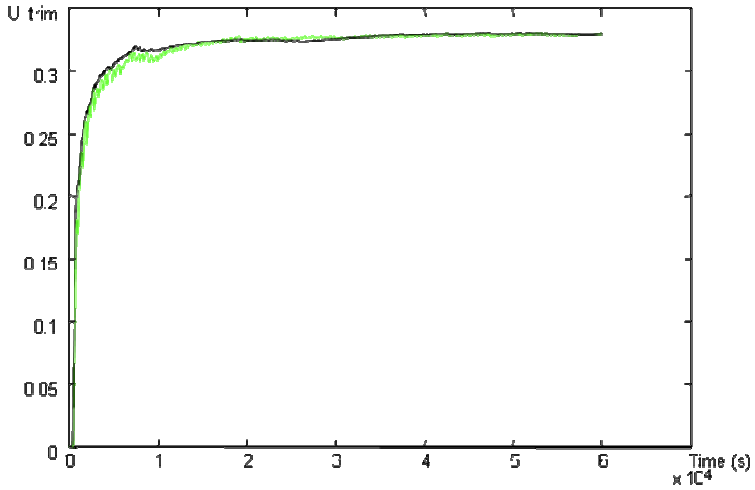

Figure 11. Utilization rate of the trimmer – Poisson process of mean 30 s

(black: complete model; gray: reduced model)

These results show that a reduced model gives similar results than a complete one. Therefore, it is relevant to use reduced model instead of complete one because it is quicker to construct and use without loss of precision.

5. CONCLUSION

A new approach for simulation model reduction has been presented here. This approach uses a neural network and, more particularly, a multilayer perceptron to model the functioning of a part of the process that is not constrained in capacity. This approach has been applied to the modelisation of a sawmill workshop. The results show that:

- the two datasets present similar results,
- the average of the error is small relative to the process time scale, and
- the complete and reduced models gave similar results even if the log arrival rule is changed.

This means that it seems efficient to use a neural network to model a part of a process instead of constructing the complete model.

Assuming that the construction of a neural network is a quasi-automated task, in which the modeler only collects and selects the input data set. It is faster and easier to construct this kind of reduced model. This approach allows the modeler to focus on the management of bottlenecks.

Our intentions for future work are to investigate the structure determination of the neural network, particularly the choice of its inputs, and the validation of this approach on different applications, particularly on several external supply chains, such that at least one particular enterprise belongs to different supply chains.

6. REFERENCES

Akyol, D.E., Bayhan, G.M. 2007. A review on evolution of production scheduling with neural networks. *Computers and Industrial Engineering*, 53, 95–122.

Aldanondo M., Mercé C., 1991. Hierarchical data model for scheduling and monntoring in manufacturing systems. In *Computer Applications in Production and Engineering: Integration Aspects, G. Doumeingts, J Browne and M. Tomljanovitch Ed., Elsevier Science Publishers B.V. North Holland*, 477-484.

Araz, O.U., Eski, O.; Araz, C., 2008. Determining the parameters of dual-card kanban system: an integrated multicriteria and artificial neural network methodology. *Int. Adv. Manuf. Technol*, 38, 965–977.

Barton, R.R., 1994. Metamodeling: A state of the art review. *Proc. of the 1994 Winter Simulation Conference*, 237–244.

Branke, J., 1995. Evolutionary algorithms for neural network design and training. *Proc. of the 1st Nordic Workshop on Genetic*, http://www.aifb.uni-karlsruhe.de/~jbr/Papers/GaNNOverview.ps.gz.

Brooks, R.J., Tobias, A.M., 2000. Simplification in the simulation of manufacturing systems. *Int. J. Prod. Res.*, 38(5), 1009–1027.

Cakar, T., Cil, I., 2004. Artificial neural networks for design of manufacturing systems and selection of priority rules. *Int. J. of Computer Integrated Manufacturing*, 17(3), 195–211.

Chamber, M., Mount-Campbell, C.A., 2002. Process optimization via neural network metamodeling. *Int. J. Production Economics*, 79, 93–100.

Chen, T., Wang, Y.C., 2009. A nonlinear scheduling rule incorporating fuzzy-neural remaining cycle time estimator for scheduling a semiconductor manufacturing factory—a simulation study. *Int. J. Adv. Manuf. Technol.*, 45, 110–121.

Chentouf, R., Jutten, C., 1996. Combining sigmoids and radial basis function in evolutive neural architecture. *European Symp. on Artificial Neural Network ESANN'96*, Bruges, Belgium, 129–134.

Chiu, M., Lin., G. 2004. Collaborative supply chain planning using the artificial neural network approach. *J. of Manufacturing Technology Management*, 15(8), 787–796.

Chwif, L., Paul, R.J., Pereira Barretto, M.R., 2006. Discrete event simulation model reduction: A causal approach. *Simulation Modeling Practice and Theory,* 14, 930–944.

Cybenko, G., 1989. Approximation by superposition of a sigmoidal function. *Math. Control Systems Signals,* 2(4), 303–314.

Doumeingts G., Vallespir B., Darricau D., Roboam M., 1987. Design methodology for advanced manufacturing systems. *Computers in Industry,* 9(4), 271-296.

Drucker, H., 2002. Effect of pruning and early stopping on performance of a boosting ensemble. *Computational Statistics and Data Analysis,* 38, 393–406.

El Bouri, A., Shah, P. 2006. A neural network for dispatching rule selection in a job shop, *Int. J. Adv. Manuf. Technol.*, 31, 342–349.

El Haouzi, H., Pétin J.F., Thomas A., 2008. Design and validation of a product-driven control system based on a six sigma methodology and discrete event simulation. *Production Planning and Control*, 20(6), 510–524.

Fang, L., Li, T., 1990. Design of competition based neural networks for combinatorial optimization. *Int. J. of Neural Systems*, 1(3), 221–235.

Feng, S., Li, L., Cen, L., Huang, J., 2003. Using MLP networks to design a production scheduling system. *Computer and Operations Research.*, 30, 821–832.

Fonseca, D.J., Navarese, D., 2002. Artificial neural networks for job shop simulation. *Advanced Engineering Informatics*, 16, 241–246.

Foo, Y.P.S., Takefuji, Y., 1988. Integer linear programming neural networks for job-shop scheduling. *Proc. of Joint Int. Conf. on Neural Networks,* 2, 341–348.

Foo, Y.P.S., Takefuji, Y., Szu, H. 1995. Scaling properties of neural networks for job-shop scheduling. *Neurocomputing,* 8, 79–91.

Forrester J.W. 1961. *Industrial dynamics*. MIT press, Cambridge, MA.

Funahashi, K., 1989. On the approximate realization of continuous mapping by neural networks. *Neural Networks,* 2, 183–192.

Frantz F.K. 1995. A taxonomy of model abstraction techniques. *Proc. of the 1995 Winter Simulation Conference*, Washington, USA.

Geneste, L., Grabot, B., 1997. Implicit versus explicit knowledge representation in a job shop scheduling decision support system. *Int. J. of Expert Systems*, 10(1), 37–52.

Goldratt, E., Cox, J., 1992. *The goal: A process of ongoing improvement*. North River Press; 2nd Revised edition, Great Barrington, USA.

Hassibi, B., Stork, D.G. 1993 Second order derivatives for network pruning: optimal brain surgeon. *Advances in Neural Information Processing Systems*, S.H. Hanson, J.D. Cowan and C.L. Gilles (Eds.), Morgan Kaufmann, San Mateo, CA, 164–171.

Ho, Y.C., 1987. Performance evaluation and perturbation analysis of discrete event dynamics systems. *IEEE Trans. on Automatic Control,* 32(7), 563–572.

Hung, Y.F., Leachman, R.C., 1999. Reduced simulation models of wafer fabrication facilities. *Int. J. Prod. Res.,* 37, 2685–2701.

Hwang, J.S., Hsieh, S., Chou, H.C., 1999. A Petri net based structure for AS/RS operation modeling. *Int. J. Prod. Res.,* 36, 3323–3346.

Innis, G.S., Rexstad, E., 1983. Simulation model simplification techniques. *Simulation,* 41, 7–15.

Kleijnen, J.P.C. 2005. Supply chain simulation tools and techniques: A survey. *Int. J. of Simulation and Process Modeling*, 1, 82–89.

Kleijnen, J.P.C., Sargent, R.G., 2000. A methodology for fitting and validating metamodels in simulation. *European J. of Operational Research*, 120, 14–29.

Kleijnen, J.P.C., Smits, M.T. 2003. Performance metrics in supply chain management. *J. of Operational Research Society*, 54, 507–714.

Khosravi, A., Creighton, D., 2010. A prediction interval-based approach to determine optimal structures of neural network metamodels. *Expert Systems with Applications*, 37, 2377–2387.

Kuo, Y., Yang, T., Peters, B.A., Chang, I., 2007. Simulation metamodel development using uniform design and neural networks for automated material handling systems in semiconductor wafer fabrication. *Simulation Modelling Practice and Theory*, 15, 1002–1015.

Kutschenreiter-Praszkiewicz, I., 2008. Application of artificial neural network for determination of standard time in machining. *J. Intell. Manuf.*, 19, 233–240.

Law, A.M., Kelton, W.D. 2000. *Simulation modeling and analysis*. Third edition, McGraw-Hill, Boston, USA.

Leachman, R.C., 1986. *Preliminary design and development of a corporate level production planning system for the semi conductor industry*, Eds Optimization in industry, Chichester, UK.

Li, J., Meerkov, S.M., Zhang, L. 2009. Production systems engineering: Problems, solutions, and application, *13th IFAC Symp. on Information Control Problems in Manufacturing INCOM'09*, Moscow, Russia, June 3–5, 1–14.

Lin, D.Y., Hwang, S.L., 1999. Use of neural networks to achieve dynamic task allocation: a flexible manufacturing system example, *Int. J. of Industrial Ergonomics*, 24, 281–298.

Mouelhi-Chibani, W., Pierreval, H., 2010. Training a neural network to select dispatching rules in real time, *Computer and Industrial Engineering*, 58, 249–256.

Musselman, K.J., 1993. Guideline for simulation project success. *Proc. of the 1993 Winter Simulation Conference*, 58–64.

Nguyen, D., Widrow, B., 1990. Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights. *Proc. of the Int. Joint Conf. on Neural Networks IJCNN'90*, 3, 21–26.

Page, E.H., Nicol, D.M., Balci, O., Fujimoto, R.M., Fishwick, P.A., L'Ecuyer, P., Smith, R., 1999. An aggregate production planning framework for the evaluation of volume flexibility. *Proc. of the 1999 Winter Simulation Conference*, 1509–1520.

Pidd, M., 1996. Five simple principles of modeling. *Proc. of the 1996 Winter Simulation Conference*, 721–728.

Pierreval, H., 1996. A metamodel approach based on neural networks. *Int. J. in Computer Simulation*, 6(3), 365–378.

Rivals, I., Personnaz, L. 2003 Neural network construction and selection in nonlinear modeling. *IEEE Trans. on Neural Networks*, 14(4), 804–819.

Setiono, R., Leow, W.K., 2000. Pruned neural networks for regression. *Int. Conf. on Artificial Intelligence PRICAI'00, Melbourne, Australia*, 500–509.

Shervais, S., Shannon, T.T., Lendaris, G.G., 2003. Intelligent supply chain management using adaptive critic learning. *IEEE Trans. on Systems, Man and Cybernetics, Part A Systems and Humans*, 33(2), 235–244.

Sukthomya, W., Tannock, J., 2005. The training of neural networks to model manufacturing processes. *J. of Intelligent Manufacturing*, 16, 39–51.

Suri, R., Fu, B.R., 1994. On using continuous flow lines to model discrete production lines. *Discrete Event Dynamic Systems*, 4, 129–169.

Thierry, C., Thomas, A., Bel, G., 2008. *Simulation for supply chain management*. John Wiley & Sons, London, UK.

Thomas, A., Charpentier, P., 2005. Reducing simulation models for scheduling manufacturing facilities. *European J. of Operational Research*, 161(1), 111–125.

Thomas, P., Bloch, G., 1996. From batch to recursive outlier-robust identification of non-linear dynamic systems with neural networks. *Proc. of the IEEE Int. Conf. on Neural Networks ICNN'96*, Washington D.C., USA, 1, 178–183.

Thomas, P., Bloch, G., 1997. Initialization of one hidden layer feedforward neural networks for non-linear system identification. *Proc. of the 15th IMACS World Congress on Scientific Computation, Modelling and Applied Mathematics WC'97*, Berlin, Germany, August 25-29, 295–300.

Thomas, P., Bloch, G., Sirou, F., Eustache, V., 1999. Neural modeling of an induction furnace using robust learning criteria. *J. of Integrated Computer Aided Engineering*, 6(1), 5–23.

Thomas, P., Choffel, D., Thomas, A., 2008. Simulation reduction models approach using neural network. *10th Int. Conf. on Computer Modeling and Simulation EUROSIM'08,* Cambridge, Great Britain, 1–3 April.

Tseng, T.Y., Ho, T.F., Li, R.K. 1999. Mixing macro and micro flowtime estimation model: Wafer fabrication. *Int. J. of Production Research*, 37, 2447–2461.

Vollmann, T.E., Berry, W.L., Whybark, D.C., 1992. *Manufacturing, Planning and Systems Control*. The Business One Irwin.

Vosniakos, G.C., Tsifakis, A., Bernardos, P., 2006. Neural network simulation metamodels and genetic algorithms in analysis and design of manufacturing cells. *Int. J. Adv. Manuf. Technol.*, 29, 541–550.

Ward, S.C., 1989. Argument for constructively simple models. *J. of the Operational Research Society,* 40(2), 141–153.

Willems, T.M., Brandts, E.M.W., 1995. Implementing heuristics as an optimization criterion in neural networks for job-shop scheduling. *J. of Intelligent Manufacturing*, 6, 377–387.

Yang, F., 2010. Neural network metamodeling for cycle time-throughput profiles in manufacturing. *European J. of Operational Research*, 205, 172–185.

Zeigler, B.P., 1976. *Theory of modeling and simulation*. Wiley, New York.

Zhang, H.C., Huang, S.H., 1995. Application of neural networks in manufacturing: a state-of-the-art survey. *Int. J. Prod. Res.*, 33(3), 705–728.