ELSEVIER

European Journal of Operational Research 145 (2003) 14–30

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

www.elsevier.com/locate/dsw

Discrete Optimization

# Makespan estimation in batch process industries: A comparison between regression analysis and neural networks

W.H.M. Raaymakers, A.J.M.M. Weijters *

*Department of Technology Management, Eindhoven University of Technology, P.O. Box 513, NL-5600 MB Eindhoven, Netherlands*

Received 29 May 2000; accepted 11 January 2002

**Abstract**

Batch processing is becoming more important in the process industries, because of the increasing product variety and the decreasing demand volumes for individual products. In batch process industries it is difficult to estimate the completion time, or makespan, of a set of jobs, because jobs interact at the shop floor. We assume a situation with hierarchical production control consisting of a planning level and a scheduling level. In this paper we focus on the planning level. We use two different techniques for estimating the makespan of job sets in batch process industries. The first technique estimates the makespan of a job set by developing regression models, the second technique by training neural networks. Both techniques use aggregate information. By using aggregate information the presented techniques are less time consuming in assessing the makespan of a job set compared with methods based on detailed information.

Tests on newly generated job sets showed that both techniques are robust for changes in the number of jobs, the average processing time, a more unbalanced workload and for different resource configurations. Finally, the estimation quality of the neural network models appears significantly better than the quality of regression models.


*Keywords:* Scheduling theory; Regression; Neural networks

## 1. Introduction

Batch processing is frequently found in food, specialty chemicals and pharmaceutical industries, where production volumes of individual products do not justify continuous production and dedicated production lines. Nowadays, batch process industries have become more important because of the increasing product variety and decreasing demand volumes for individual products. Two basic types of batch process industries are distinguished. If products all follow the same routing, this is called *multiproduct*. If products follow different routings, like in a traditional job shop, it is called *multipurpose*. In this paper, we concentrate on multipurpose batch process industries.

Multipurpose batch process industries are inherently the most flexible and complex: products may differ in the number, type, and duration of processing steps. Intermediate products are often unstable, which means that after processing some steps, a product needs to be processed further

* Corresponding author. Tel.: +31-40-247-3857; fax: +31-40-243-2612.

*E-mail address:* a.j.m.m.weijters@tm.tue.nl (A.J.M.M. Weijters).

0377-2217/03/$ - see front matter © 2002 Published by Elsevier Science B.V.
PII: S0377-2217(02)00173-X

without any delay. These no-wait restrictions and the large variety of products cause complex scheduling problems. Consequently, the capacity utilization realized by multipurpose batch process industries is generally low. Furthermore, the amount and mix of production orders may differ considerably from period to period due to variability and dynamics of demand. Consequently, bottlenecks may shift over time.

One of the main difficulties for this type of industries is to estimate the workload that can be completed during a specific period (Raaymakers et al., 2000). The capacity utilization that can be realized strongly depends on the mix of production orders (jobs). In order to set reliable due dates, it is important for planners to accurately estimate what workload and mix of jobs can be completed in a specific period. If planners can accurately estimate the workload and mix that can be completed, they can obtain achievable production plans. In this paper we consider a situation in which planners periodically release a set of jobs to the shop floor.

The makespan estimate of a job set is used to assess whether a job set can be completed within a given period. A job set is achievable if a schedule can be constructed in which all jobs are completed before the end of the period. This means that the makespan of the schedule is smaller or equal to the period length. The makespan of a schedule is the time at which the last job in the job set is completed.

In this paper, we use two techniques for estimating the makespan of job sets based on aggregate characteristics. The first technique estimates the makespan by using a regression model. The second technique uses neural network models to estimate the makespan. The main advantage of the regression models is that these provide insight into the relation between job set characteristics and the makespan. However, the possible relations between them have to be specified prior to performing the regression analysis. The main advantage of artificial neural networks is that these can automatically detect complex non-linear relations between the job set characteristics and the makespan. Therefore, artificial neural networks may provide more accurate estimates. On the other hand, interpretation of models induced by neural networks is often extremely difficult or impossible due to the non-linear and non-symbolic nature of the models.

The structure of this paper is as follows. In Section 2, we present our problem definition. In Section 3, we present the data set used for comparison of the techniques. Section 4 starts with an overview of related research (Section 4.1). Then, two different techniques for makespan estimation are presented: regression models (Section 4.2) and neural network models (Section 4.3). In Section 5, the estimation quality of both types of models is evaluated and compared. Finally, Section 6 summarizes our conclusions.

## 2. Problem definition

### 2.1. Planning procedure

We assume a situation with hierarchical production control consisting of a planning level and a scheduling level. At the planning level, the planner is responsible for order acceptance and assigning the accepted orders to time periods (e.g. weeks). The set of orders assigned to each time period should be achievable, which means that a schedule can be constructed that completes all assigned production orders (=jobs) within that period. At the scheduling level, the scheduler constructs a detailed schedule for the job set that is assigned to the current period. This schedule is then executed at the shop floor. The planning horizon of the scheduler is much shorter than the planning horizon of the planner.

Scheduling industrial job sets showed that the workload that can be completed in a specific planning period depends on the current mix of jobs (Raaymakers and Hoogeveen, 2000). This complicates the work of the planner, who has to determine which jobs are assigned to a specific period. He has to consider both workload and job mix characteristics in order to determine achievable job sets.

The planner may evaluate the achievability of a job set for a planning period by constructing a schedule. This would result in constructing a detailed schedule over a medium term horizon. The main disadvantage of this approach is that it is time consuming because a new schedule has to be

constructed every time a change in the plan occurs, for example when a new customer order arrives. Furthermore, the constructed schedule is of little value, because disturbances at the shop floor or changes in the job sets are likely to occur. Assigning jobs to periods can be part of order acceptance and due date setting. These decisions generally need to be made a number of periods before the actual execution of the jobs at the shop floor. In the meantime changes in production and demand are likely to occur. Therefore, we prefer methods that assess the achievability of job sets based on simple characteristics of the jobs and the available resources.

### 2.2. Definition of production characteristics

The job sets and resource sets considered in this paper originate from multipurpose batch process industries. Typical for this type of industries is that jobs differ in the number of processing steps, the resource types required, the sequence of processing steps and the processing times. In that sense, these industries are similar to discrete job shops. However, there are two main differences: (1) several resources may be needed at the same time, and (2) no-wait restrictions exist. For example, a chemical reaction needs to be followed immediately by a filtration and then by a step in which the filtrated intermediate is dried, because the intermediate is unstable. The total of the reaction, the filtration and the drying is then considered as one job. For this job a number of resources is required: a reaction vessel, a filter, a vessel to keep the fluid that goes through the filter and a drying resource. These resources are required partly in parallel. An illustration of a job is given in Fig. 1: resource type I is a (reaction) vessel, resource type II a filter, and resource type III drying equipment. In our study we have modeled processing steps in such a way that each processing step requires exactly one resource and that processing steps may need to be performed in parallel. The fixed time delay then reflects the no-wait restrictions.

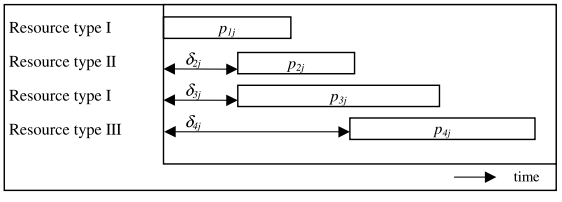


Fig. 1. Job definition.

For a period a job set consisting of $J$ jobs has to be completed on a given set of resources. Jobs are defined as follows. Each job ($j$) consists of a specific number of processing steps ($s_j$). These processing steps may have an overlap in time. The start time of each processing step is given by the time delay ($\delta_{ij}$) and fixed. Also, processing times ($p_{ij}$) are given for each step. The jobs have to be completed on $N$ resources of $M$ different types. To complete the job given in Fig. 1, four resources of three different types are required.

The following assumptions regarding jobs and resources are made:

- All jobs are available at the start of the period.
- Resources are available from the start of the period and without interruptions.
- Resources of the same type are assumed to be identical.
- No precedence relations exist between jobs.
- Each processing step has to be performed without pre-emption on exactly one resource of a specific resource type.
- More than one processing step of a job may require the same resource type. These processing steps have to be performed on different resources of that type if they overlap.
- Set up times are included in the processing times and are sequence independent.
- Processing times are given (deterministic).

### 2.3. Relation between planning and scheduling

In our study we compare two techniques that can be used by the planner to predict the time required to complete a specific job set (= makespan) based on aggregate characteristics of the job set. With these techniques, the planner can predict the achievability of a specific job set, i.e. whether the predicted makespan is not longer than the planning period.

The number of jobs that can be assigned to a given planning period by the planner depends on

the ability of the scheduler to construct a good schedule. If the scheduler is able to construct schedules that result in a high utilization of the resources, then the planner can assign a higher workload to a planning period. Consequently, the prediction techniques that we studied depend on the scheduling techniques used.

The scheduling situation considered in this study can be characterised as a no-wait job shop. For larger values of $J$, we cannot expect to find an optimal solution in reasonable time because no-wait job shop scheduling problems are NP-hard (Lenstra et al., 1977). Therefore, we have chosen simulated annealing to obtain solutions to the scheduling problems. A simulated annealing procedure that aims at minimization of the makespan and that obtains near-optimal solutions has been implemented and tested for industrial instances (Raaymakers and Hoogeveen, 2000). Minimum makespan corresponds to minimum idle time on the resources and hence, maximum overall capacity utilization. In our specific case, it is difficult to determine the quality of the simulated annealing results, since there are no tight lower bounds available for this specific scheduling problem.

### 2.4. Job interaction at the scheduling level

The workload that can be completed in a period depends on the mix of jobs that are assigned to that period. We argue that this is caused by job interactions at the scheduling level. Job interactions occur because each job requires one or more resources at the same time or consecutively without delay. Due to no-wait restrictions, idle time on the resources can generally not be avoided. Consequently, job interactions result in idle time on the resources and hence to an increase in the makespan of a job set. The amount of job interaction depends on the job mix and the resource set of a production department. Therefore, we use aggregate characteristics of the jobs and the resources to estimate the amount of job interaction. This estimate is then used to estimate the makespan of a job set.

The amount of interaction is measured by the interaction margin ($I$). The interaction margin is defined as *the relative difference between the makespan realized while meeting all no-wait restrictions and a lower bound on the makespan*:

$$I = \frac{C_{\max} - lb}{lb}, \tag{1}$$

where $I$ is the interaction margin; $C_{\max}$ is the makespan or completion time of the job set; and $lb$ is the lower bound on the makespan.

The lower bound used in this paper is equal to the amount of time required to complete all jobs if the total processing time can be distributed evenly over the resources of the bottleneck resource type, and if these resources can all process without interruption. If a feasible schedule is found which has no idle time on any resource of the bottleneck resource type, then the makespan is equal to this lower bound. The interaction margin is, by definition, equal to zero in that situation. In all other situations, where the makespan is higher than the lower bound, idle time exists on the bottleneck resource type.

For a specific job set the total workload on each resource type is obtained by summing the processing times of all processing steps that require that resource type. The bottleneck resource is the resource type with the maximum of the workload divided by the number of resources of that type. A lower bound ($lb$) on the makespan is obtained by dividing the workload on the bottleneck resource by the number of resources of that resource. We may round this up because all processing times are integer values and no pre-emption is allowed:

$$lb = \lceil \max\{L_1/n_1, L_2/n_2, \ldots, L_M/n_M\} \rceil, \tag{2}$$

where $M$ is the number of resource types, $n_m$ is the number of resources of resource type $m$, $m = 1, \ldots, M$, and $L_m$ is the workload on resource type $m$, $m = 1, \ldots, M$.

In this paper, we focus on makespan estimation using five variables that influence the amount of interaction. Previous work (Raaymakers and Fransoo, 2000) showed that these variables explain a large part of the variation in the amount of interaction. These variables are:

- average number of parallel resources ($\mu_a$),
- average number of processing steps per job ($\mu_s$),
- average overlap of processing steps ($\mu_g$),

- workload balance ($\rho_{\max}$), and
- standard deviation in the processing time ($\sigma_p$).

The workload balance ($\rho_{\max}$) is the average utilization of the resources that is realized if the job set is being processed without any idle time on the bottleneck. This means that when $\rho_{\max}$ is close to 1, all resource types are equally loaded. A lower $\rho_{\max}$ means that (considerable) idle time on some resource types cannot be avoided, because they have a lower load than the bottleneck resource.

As mentioned in Section 2.3, the processing times for each processing step are given and deterministic. The standard deviation in the processing time ($\sigma_p$) thus gives the variation in processing times within the total job set. A small $\sigma_p$ indicates that the duration of the processing steps in a job set are more or less the same. A large $\sigma_p$ indicates that there is a mix of shorter and longer processing times in the job set.

## 3. Experimental data

To compare both techniques for makespan estimation, we need to have a data set of job sets with their characteristics and the makespan of the schedule constructed. To obtain such data set, we have randomly generated a number of job sets that resemble realistic job sets from industry. There are two reasons for using randomly generated job sets instead of industrial job sets. The first reason is that only a very limited number of job sets could be obtained from industry. The second reason is that by generating job sets we can influence the characteristics of these job sets by setting the levels as shown in Table 1. These parameters are set such that we can better observe the effect of the interaction variables and their effect on each other. An overview of the levels of the interaction variables that are used in generating the job sets is given in Table 1.

The values of the interaction variables are varied on three levels for the processing time, four levels for the number of parallel resources, and five levels for the number of processing steps and the overlap of processing steps. The workload balance ($\rho_{\max}$) is not varied in the experiments because it is a random factor. In each experiment a job set containing 50 jobs is generated that have to be completed on 10 resources. To test the influence of the average number of parallel machines, we consider four different resource sets (A, B, C and D).

For each job, the number of processing steps is given. The number of processing steps per job cannot exceed the number of resources because each processing step of a job has to be performed on a different machine if the processing steps are overlapping in time. To each processing step a resource type is allocated at random. The probability of a resource type being allocated is equal to the number of resources of that type available related to the total number of resources available. The number of processing steps of a specific job that are allocated to the same resource type cannot exceed the number of resources of that specific type. In this way each resource type has equal probability to become a bottleneck resource. As a result the workload balance of the resource types is kept high.

A processing time is then allocated to each processing step. In the experiments the average

Table 1
Levels used in the computer experiments to generate the data set for building the makespan estimation models

| Interaction variables | Levels used in generating job sets |
|---|---|
| Number of parallel resources | A = 10 types, each 1 resource<br>B = 5 types, each 2 resources<br>C = 2 types, each 5 resources<br>D = 1 type, 10 resources |
| Number of processing steps | All jobs in a job set 1, 3, 5, 7, 10 steps |
| Overlap of processing steps | All jobs in a job set overlap of 0, 1/4, 1/2, 3/4, 1 |
| Processing time distribution | 25, Uniform (15–35), uniform (1–49) |

processing time is kept constant at 25. Finally, the time delay of each processing step is determined. The time delay of the first processing step is zero for each job. The second and following processing steps start when a given overlap of the previous processing step has been processed. An overlap of 0 means that all processing steps are executed consecutively; whereas an overlap of 1 means that all processing steps start at the same time.

A full factorial experimental design is constructed. As noticed above, the workload balance cannot be varied on distinct levels because it is the result of the random allocation of processing times and resource types to processing steps. Therefore, this predictor variable is not included in the experimental design. For each combination five replications have been carried out. A near-optimal schedule with respect to makespan is obtained by using Simulated Annealing (Raaymakers and Hoogeveen, 2000).

### 3.1. Simulation results

In total, 1176 job sets have been generated and scheduled. The average and standard deviation of the interaction margin are given in Table 2. On the first row the results for all resource configurations together are given. The other rows give the results for the four resource configurations separately. The average and standard deviation of the interaction margin are largest for configuration A and smallest for configuration D. This suggests that an increase in the number of identical resources results in a decreasing interaction margin. Intuitively, this can be explained as follows. An increase in the number of identical resources results in more scheduling flexibility, and hence to a decreasing job interaction.

Table 2
Interaction margin per resource configuration

| | $\bar{I}$ | S.D. ($I$) |
|---|---|---|
| Total | 0.49 | 0.38 |
| Resource configuration A | 0.58 | 0.48 |
| Resource configuration B | 0.58 | 0.40 |
| Resource configuration C | 0.43 | 0.25 |
| Resource configuration D | 0.30 | 0.17 |

The influence of the average number of processing steps, the average overlap and the variation in processing times on the interaction margin is given in Figs. 2–5. In these figures the results are given separately for each of the resource configurations. The interpretation of these figures should be done with care, because they give the aggregated results of the 1176 experiments. Interaction between different variables cannot be concluded from these figures.

Fig. 2 shows that the average interaction margin increases with an increase in the average number of processing steps. This increase is strongest for resource configuration A and weakest for resource configuration D.

Fig. 3 shows the influence of the average overlap on the interaction margin. The average interaction margin decreases with an increase in the overlap of processing steps. An increasing overlap results in a decrease in the time between start and completion of a job, if all other job parameters remain unchanged. Therefore, it is not surprising that an increasing overlap results in a decreasing

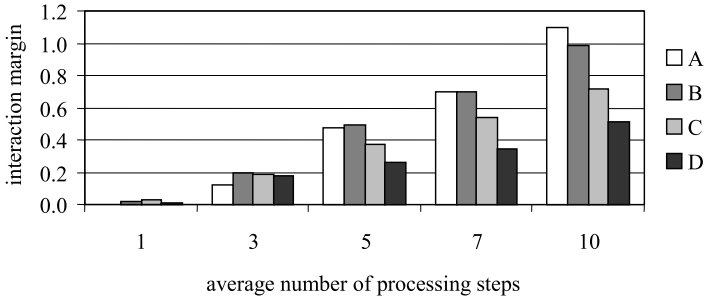


Fig. 2. Influence of average number of processing steps on the interaction margin.

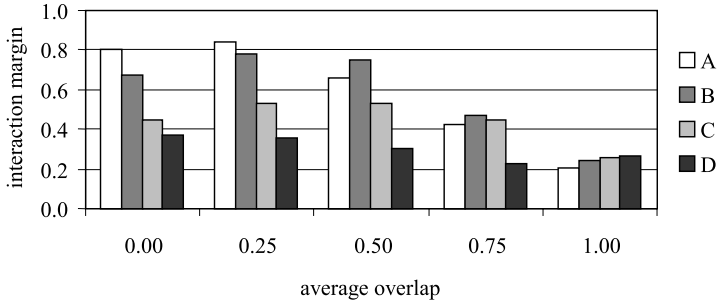


Fig. 3. Influence of average overlap on the interaction margin.

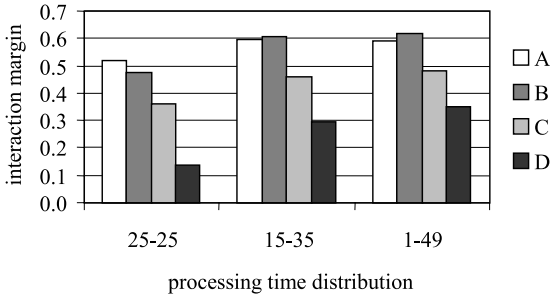


Fig. 4. Influence of processing time distribution on the interaction margin.

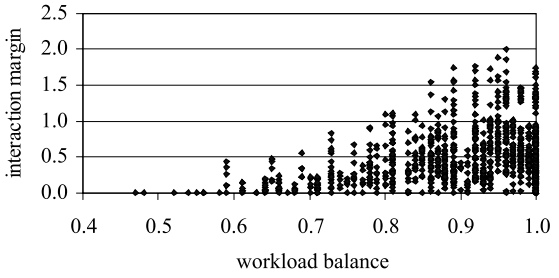


Fig. 5. Influence of workload balance on the interaction margin.

makespan for the job set, and hence to a decreasing interaction margin. Here a difference also exists between the resource configurations. For an overlap of 0 resource configuration A has the highest average interaction margin, whereas for an overlap of 1 the difference between resource configurations is small.

The influence of the standard deviation of the processing time seems to have a smaller impact on the interaction margin. Fig. 4 shows that an increase in the standard deviation of the processing time results in an increase in the average interaction margin.

Fig. 5 shows the influence of the workload balance on the interaction margin. As mentioned earlier, the workload balance is the result of random allocations of processing times and resource types to processing steps. Consequently, the value of the workload balance cannot be varied on distinct levels. In the experiments, the value of the workload balance was between 0.45 and 1.

In this section, only an indication of the relations between the different interaction variables and the interaction margin is given. However, to use these relations for estimating the interaction margin for job sets we need to quantify the relations.

## 4. Makespan estimation

In this section, we build models that can be used to estimate the interaction margin of a job set based on the five interaction variables: average number of identical resources ($\mu_a$), the average number of processing steps ($\mu_s$), the average overlap ($\mu_g$), the standard deviation of the processing times ($\sigma_p$), and the workload balance ($\rho_{\max}$). The job sets discussed in the previous section are used for building the models. The section starts with an overview of related research (Section 4.1). Then, two different techniques for makespan estimation are presented: regression models (Section 4.2) and neural network models (Section 4.3). The

estimation quality of both types of models is evaluated and compared in the next section (Section 5).

### 4.1. Previous research

Makespan estimation is related to flowtime estimation, which has received considerable attention in the literature on due date assignment. For an overview, we refer to Cheng and Gupta (1989). The flow time is the total throughput time of a job in a production system, which consists of processing time and waiting time. Flowtime estimation is generally used in the determination of achievable due dates for customer orders. Upon arrival of a customer order the flow time is estimated for the jobs related to that customer order. The due date of the job is set to the arrival date plus a flow allowance, which is the estimated flow time. Several rules for determining flow time allowances are proposed in the literature. A distinction can be made between rules that use job characteristics only, and rules that also use shop information. A simple and popular rule that uses job characteristics only is the total work-rule. According to this rule, the flow time allowance is proportional to the total processing time of the job (Eilon and Chowdhury, 1976; Weeks, 1979; Baker and Bertrand, 1981a,b). Bertrand (1983) introduced a rule that uses both the total processing time of a job and the number of processing steps to determine the flow time allowance. Other rules also use information on the current shop status, which is generally viewed as a queuing network. Some rules use the expected waiting time per job or per operation as a basis for flow time estimation (Eilon and Chowdhury, 1976; Weeks, 1979). Also, the number of jobs in the queues can be used for flow time estimation. Some of these rules use information on the total workload or total number of jobs in the shop, while others use only information of the resources on the job's routing. Bertrand (1983) introduced the use of time-phased workload to estimate the flow time of a job. Vig and Dooley (1991) introduced the use of the average flow time per operation of three recently completed jobs to estimate the flow time of a new job. The different job characteristics and shop congestion characteristics may also be used in combination to determine the flow time allowance. Ragatz and Mabert (1984) and Vig and Dooley (1991, 1993) use regression analysis to determine the weighting of coefficients for the different job and shop characteristics included in the flowtime estimations. The characteristics included, and the corresponding coefficients, depend on the dispatching rule that is used for the execution of the jobs on the resources in the shop. Vig and Dooley (1993) showed that using combined static and dynamic flowtime estimation methods yield flowtime estimates that are more accurate and more robust to variable shop condition, than realized by dynamic flowtime estimation methods. Enns (1993, 1995, 1998) evaluates a number of dispatching rules and due date settings in a simulation study. Appealing to his approach is that he distinguishes between internal and external due dates. Internal due dates are based on the estimated flow time and are used for dispatching. External due dates include the estimated flow time and the estimation error on this flow time in order to realize a specific delivery reliability to the customer. A similar approach is chosen by Lawrence (1995).

A comparison of different rules shows that using information on the shop status improves the flow time estimation (Eilon and Chowdhury, 1976; Weeks, 1979). Furthermore, using information on the number of jobs or the workload in the queues along the job's routing performs better than using general shop congestion information (Ragatz and Mabert, 1984; Vig and Dooley, 1991). The performance of the flow time estimation rules is influenced by the dispatching rule used in the shop.

A number of authors use neural networks for flow-time estimation (Arizono et al., 1992) or job (shop) scheduling (Hill and Remus, 1994; Lee and Kim, 1993; Statake et al., 1994). Sabuncuoglu (1998) presents a review of the literature and new research directions of scheduling approaches with neural networks. Job-shop scheduling problems are considered as combinatorial optimization problems and appropriate neural network architectures (e.g. a Hopfield network or the Gaussian machine model) are used for minimizing the total actual flow time. Although the neural network

approach has the possibility of solving various kinds of scheduling problems, how to specify the values of many parameters and weights of these networks remains a critical issue.

Sabuncuoglu and Gurgun (1996) combine algorithmic and neural net approaches to solve the single machine mean tardiness scheduling problem and the minimum makespan job-shop scheduling problem. Chen and Muraki (1997) use a standard back-propagation neural network for on-line rescheduling on the basis of preprocessed information about the plant status. Finally, Philipoom et al. (1994) compare non-linear regression models and neural networks for due-date assignments for job scheduling. An important difference between the mentioned neural network approaches and our approach is that our study focuses on the *makespan of job sets* as the basis of information aggregation. In this sense there is a certain analogy between our approach and the neural network based approach of Huang et al. (1999); they present a successful neural network production performance model based on only two careful selected input variables.

### 4.2. Makespan estimation using regression models

In this section we use regression models to build makespan estimation models. The motivation to use regression models is that they provide insight into the relations between the interaction margin and the interaction variables. Before performing the regression analysis we have to specify the possible relations between the interaction variables and the interaction margin. Raaymakers and Fransoo (2000) and Raaymakers et al. (2001) showed that both main effects and two-way interactions of the defined interaction variables influence the interaction margin significantly. Therefore, both main effects and two-way interactions are considered in developing the estimation models. Alternative estimation models are generated by means of regression analysis, based on these five interaction variables. The variables used in the regression analysis are called the predictor variables, whereas the interaction margin is called the response variable. Before performing the regression analysis this data was checked on outliers and orthogonality of the predictor variables. These results show that there exists a considerable correlation between $\rho_{\max}$ on the one hand, and $\mu_a$ and $\mu_s$ on the other hand. Therefore, we have to be aware of multicollinearity, which occurs when there is a correlation between the predictor variables. Multicollinearity can be detected by using *variance inflation factors* (VIFs) (Montgomery and Peck, 1992). In building alternative estimation models we only consider those models for further evaluation that do not have high VIFs.

First, an estimation model (RM1) is constructed that only contains the main effects of the interaction variables. Backward regression is used to eliminate variables that do not have a significant contribution. Second, estimation models are constructed that include two-way interactions by using stepwise regression. If a variable in the estimation model looses its significance, due to the inclusion of another variable, then the non-significant variable is removed from the model. The resulting estimation models are presented in Table 3. Given are the number of the model, the number of predictor variables, the predictor variables, the adjusted $R^2$, and the standard deviation of the estimate. The adjusted $R^2$ indicates the part of the variation in the interaction margin that is explained by the model. An $F$-test is used for each

Table 3
Alternative general estimation models

| | No. | Predictor variables | Adjusted $R^2$ | S.D. ($\hat{I}$) |
|---|---|---|---|---|
| RM1 | 5 | $\mu_a\ \mu_s\ \mu_g\ \sigma_p\ \rho_{\max}$ | 0.79 | 0.17 |
| RM2 | 3 | $\mu_s\ \mu_a \cdot \mu_s\ \mu_s \cdot \mu_g$ | 0.81 | 0.16 |
| RM3 | 4 | $\mu_s\ \mu_a \cdot \mu_s\ \mu_s \cdot \mu_g\ \mu_a \cdot \mu_g$ | 0.83 | 0.15 |
| RM4 | 5 | $\mu_s\ \mu_a \cdot \mu_s\ \mu_s \cdot \mu_g\ \mu_a \cdot \mu_g\ \sigma_p \cdot \rho_{\max}$ | 0.84 | 0.15 |
| RM5 | 6 | $\mu_a \cdot \mu_s\ \mu_s \cdot \mu_g\ \mu_a \cdot \mu_g\ \mu_s \cdot \rho_{\max}\ \mu_g \cdot \sigma_p\ \mu_s \cdot \sigma_p$ | 0.86 | 0.14 |

estimation model, which showed that all models are significant. The significance of the individual predictor variables is tested using a *t*-test. All variables in the models presented are significant. Details on the procedure followed can be found in Raaymakers (1999).

### 4.3. Makespan estimation using artificial neural networks

In this section, we use neural networks (NNs) to build makespan estimation models. NNs are universal function approximators and are therefore attractive for automatically learning of the (non-linear) functional relation between the five interaction variables and the interaction margin. As learning material for the NNs we use the same material, as described in the previous section, i.e. learning instances of five interaction variables (the input of the NN) and the corresponding interaction margin (the output of the NN). To model the functional relation between them we use multi-layered feedforward networks (MFNs) trained with the on-line error back-propagation (BP) learning rule (Rumelhart et al., 1986). Normally, many training epochs are required before a set of weights is found that accurately fit the training material. However, if we use to many training epochs, a NN tend to *overfit* the learning material (i.e. the accuracy on the training material is very high whereas the accuracy on new instances is much lower). In the next section we will concentrate on a sound methodology to avoid overfitting.

#### 4.3.1. The pilot experiments

In this section, we concentrate on the so-called *pilot experiments*: the goal of these experiments is to determine an optimal number of hidden units. In the next section we will concentrate on building our final NN-based makespan estimation models.

As stated above, NNs tend to overfit the learning material (i.e. the accuracy on the training material is very high whereas the accuracy on new instances is much lower). Commonly used heuristics to avoid overfitting are (i) *early stopping* (Prechelt, 1994), and (ii) minimizing the number of hidden units.

Early stopping is a common method to prevent neural networks from overfitting the learning material based on cross validation. The available material is split into three partitions. One partition, called the learning partition, is used to perform the training of the network. A second partition, called the validation partition, is used to evaluate the quality of a network during training. Finally, a third partition called the test partition is used to estimate the performance of the trained network on *new material*. To avoid overfitting, training does not proceed until a minimum of the error on the training partition is reached, but only until a minimum of the error on the validation partition is reached. It is however possible that this low performance on the validation material is a matter of luck and that the performance of the trained NN on other new material is much lower. To indicate if this is possible the case, the third partition, the test partition, is used. The estimation of the performance of the trained network is based on the accuracy of the network on the test partition. The early stopping heuristic (i.e. a validation partition) is used in all our experiments.

If the number of hidden units is too small, the modeling capacity of the network is too low, and it is impossible for the learning rule to find an adequate model. If, on the other hand, the number of hidden units is too large, the modeling capacity of the network is too substantial, resulting in a strong inclination towards overfitting. The optimal number of hidden units for our makespan estimation models is determined by performing a 10-fold-cross-validation (10-fold-cv) pilot experiment. During the performing of a 10-fold-cv experiment, the 1176 available learning instances are split into 10 subsets with about 10% of the instances. During each of the 10 pilot experiments one subset is used as validation partition, another subset as test partition, and the remaining eight subsets as training partition for the NN. In each experiment a different random initialization of the weights of the network is used (the random initialization determinates the starting point in the weight search space).

The accuracy of a trained network is measured by calculating the mean square error (MSE) on the test partition. In all our experiments, the BP

*learning rate* was set to 0.15 and the *momentum term* to 0.4. The learning rate determines the size of the steps in the search space to find the minimal training error. A small learning rate results in long learning times. A relatively large learning rate results in faster learning but can also result in a chaotically learning behavior during training of the network. The function of the momentum term is to increase the size of the optimization steps when the direction in the weight update is the same as the direction in the previous step. As with the learning rate, if the momentum term is too large the network will display a chaotic learning behavior. Because learning time is not the issue in our approach, we choose both learning parameters relatively small. The networks were trained for a fixed number of cycles ($m = 10{,}000$). The minimal MSE was always reached well within the limit $m$.

Table 4 displays the mean square error (MSE) on the test partition of the total 80 pilot experiments; for each number of hidden units (5, 10, 15, 20, 25, 30, 40, and 50) of the 10 experiments. The two bottom lines display the average and the standard deviation (S.D.) of the MSE. The average MSE of the NNs with 25 units at the hidden layer is minimal (0.0018). A MSE of 0.0018 means that the estimation performance of the NN models on the pilot test material appears to be near to perfect ($R^2$ is about 0.98). Accordingly, in the next section in which we concentrate on our truly NNs based makespan estimation models, we also use MFNs with a hidden layer of 25 units. After all, the pilot results indicate that NNs with less then 25 hidden units have not enough modeling capacity for the modeling task at hand. On the other hand, NNs with 30 hidden units or more tend towards overfitting.

No further search was made for a potentially more optimal architecture or learning parameter setting. It should be noted that these architecture and learning parameter setting are determined without any use of the ultimate test material.

#### *4.3.2. Training the neural network based makespan estimation models*

To develop our final makespan estimation models we use the MFN-architecture as determined in the previous section (five input units, one hidden layer with 25 units and one output unit) and the same learning parameters. However, the resulting NN-model depends on the random initialization of the network weights: to get insight into estimation performance of trained NNs we again use a 10-fold-cv experimental setup. We use

Table 4
Results of the 80 pilot experiments; for each numbers of hidden units (5, 10, 15, 20, 25, 30, 40, and 50 units) the mean square error (MSE) on the test partition is given

| Pilot experiment | Number of units in the hidden layer | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 5 MSE | 10 MSE | 15 MSE | 20 MSE | 25 MSE | 30 MSE | 40 MSE | 50 MSE |
| 1 | 0.0030 | 0.0015 | 0.0021 | 0.0026 | 0.0023 | 0.0018 | 0.0021 | 0.0019 |
| 2 | 0.0035 | 0.0033 | 0.0031 | 0.0031 | 0.0029 | 0.0027 | 0.0029 | 0.0032 |
| 3 | 0.0037 | 0.0031 | 0.0027 | 0.0027 | 0.0023 | 0.0026 | 0.0035 | 0.0032 |
| 4 | 0.0015 | 0.0019 | 0.0023 | 0.0012 | 0.0014 | 0.0022 | 0.0013 | 0.0016 |
| 5 | 0.0016 | 0.0012 | 0.0009 | 0.0010 | 0.0008 | 0.0012 | 0.0012 | 0.0013 |
| 6 | 0.0022 | 0.0013 | 0.0011 | 0.0018 | 0.0019 | 0.0018 | 0.0020 | 0.0012 |
| 7 | 0.0034 | 0.0033 | 0.0028 | 0.0028 | 0.0025 | 0.0032 | 0.0031 | 0.0033 |
| 8 | 0.0023 | 0.0013 | 0.0016 | 0.0013 | 0.0017 | 0.0017 | 0.0014 | 0.0017 |
| 9 | 0.0016 | 0.0011 | 0.0010 | 0.0008 | 0.0009 | 0.0009 | 0.0008 | 0.0010 |
| 10 | 0.0024 | 0.0017 | 0.0015 | 0.0016 | 0.0013 | 0.0017 | 0.0017 | 0.0014 |
| Average | 0.0025 | 0.0020 | 0.0019 | 0.0019 | 0.0018 | 0.0020 | 0.0020 | 0.0020 |
| S.D. | 0.0008 | 0.0009 | 0.0008 | 0.0008 | 0.0007 | 0.0007 | 0.0009 | 0.0009 |

The bottom lines display the average and the standard deviation (S.D.) of the MSE. The average MSE of the NN with 25 units at the hidden layer is minimal (0.19).

Table 5
Mean square estimation errors of the 10 trained NNs

| NN-model | Learning material MSE | Validation material MSE | Learning + validation MSE |
|---|---|---|---|
| 1 | 0.0016 | 0.0014 | 0.0016 |
| 2 | 0.0015 | 0.0027 | 0.0016 |
| 3 | 0.0015 | 0.0024 | 0.0016 |
| 4 | 0.0017 | 0.0012 | 0.0016 |
| 5 | 0.0017 | 0.0010 | 0.0016 |
| 6 | 0.0016 | 0.0012 | 0.0016 |
| 7 | 0.0015 | 0.0026 | 0.0016 |
| 8 | 0.0016 | 0.0013 | 0.0016 |
| 9 | 0.0017 | 0.0009 | 0.0016 |
| 10 | 0.0016 | 0.0014 | 0.0016 |
| Average | 0.0016 | 0.0016 | 0.0016 |
| S.D. | 0.0001 | 0.0007 | 0.0000 |

The two bottom lines display the average mean square error and the standard deviation (S.D.).

all the 1176 training instances to build our estimation models. The learning material is partitioned in 10 sets of 10% of the material. Ten learning experiments are performed in which one of the 10 partitions is used as validation material (when to stop learning); the remaining 90% of the material is used as training material. In each experiment a different random weight initialization is used. This results in 10 different NN-based makespan estimation models (model 1–10).

Table 5 reports the mean square estimation errors (MSEs) of the 10 trained NNs on the learning material, the validation material and the union of learning and validation material. The standard deviation of the MSE on validation material is relatively high (0.0007); the standard deviation on learning material and the union of learning and validation material are relatively low (about zero). Our main interest, however, is the performance of the NN-models on *new* material. In the next section we evaluate the estimation quality of the regression models and the NN-models for a number of newly generated job sets.

## 5. Comparison of results

The quality of a makespan estimation model depends on its estimation on *new* job sets (not used during the development of the model). In Section 5.1 we describe the newly generated material (the test material), in Section 5.2 we describe the performance indicators used for comparing the estimation quality of our models. In Section 5.3 we use these performance indicators to compare the estimation results of our models on the newly generated test material.

### 5.1. Description of the test material

To get insight into the performance of our estimation models on new material we generated eight test sets with different characteristics:

I similar characteristics as the original data set,
II variations in the interaction variables,
III variations in the number of jobs in the job set,
IV variations in the average processing time,
V an unbalanced workload,
VI a combination of the variations in the previous test sets,
VII different resource configurations consisting of 10 resources,
VIII variations in the total number of resources.

Test set I contains job sets that are obtained by using similar input parameters as for the job sets on which the models are based. Only variations in the variables in the model were allowed, and these variations are kept within the bounds set by the original data. To build the estimation models, the

variables were varied on distinct levels. In the first test set, intermediate levels are chosen for these variables.

In test set I all jobs in a job set had the same number of processing steps. Also, all jobs in a job set had the same overlap. In test set II variations in the number of processing steps and the overlap are allowed. The resource configurations and consequently the standard deviation in the number of identical resources is kept constant, throughout the first six sets. This will be varied in test sets VII and VIII. If the decision not to include these variables in the estimation models was justified, then it is expected that varying these variables does not influence the estimation quality of the models.

In test set III the assumption that the number of jobs does not influence the interaction margin is evaluated. For one half of the test set, the number of jobs was reduced to 25, for the other half the number of jobs was increased to 100. The other jobs set characteristics are identical to the characteristics of test set I.

In test set IV the assumption that the average processing time is not of influence to the interaction margin is evaluated. The average processing time is kept constant at 25 time units for all job sets in the original data set. For one half of the job sets in test set IV the average processing time was reduced to 12.5, for the other half of the job sets the average processing time was increased to 50. The other job set characteristics are identical to test set I.

In test set V, the basic constraint that each resource type has equal probability of becoming the bottleneck resource type is relaxed. This means that in creating jobs, processing steps have a higher probability to be allocated to some resource types than to others. Consequently, the workload will be less balanced over the resource types than in the original data. This means that we extrapolate $\rho_{\max}$. To generate such job sets we use different allocation probabilities for different resource types. Obviously, this cannot be done for resource configuration D, because there is only one resource type in that situation. The allocation probabilities for the original data set and this test set are given in Table 6. Recall that the resource configurations A, B, and C have 10, 5, and 2 resource types, respectively. The number of resources per resource type is the same for all resource types within the same configuration. Furthermore, the processing time distribution is also the same for each resource type. Hence, the allocation probabilities used for the original data set results in equal expected workload for each resource type.

Test set VI contains a combination of all aspects evaluated in the previous test sets. Variations in the number of processing steps and variation in the overlap are allowed. Furthermore, the number of jobs in the test set and the average processing time is varied. Also, the workload may be more unbalanced than in the original data set.

Two additional test sets are constructed to evaluate whether the models are applicable to different resource configurations. In test set VII, the number of resources remains 10. However, the distribution of the resources over resource types is different from the original job sets. In test set VIII also the total number of resources is altered. One half of the job sets have five resources, and the

Table 6
Allocation probabilities

| | Resource type | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| A orig. | 0.10 | 0.10 | 0.10 | 0.10 | 0.10 | 0.10 | 0.10 | 0.10 | 0.10 | 0.10 |
| B orig. | 0.20 | 0.20 | 0.20 | 0.20 | 0.20 | | | | | |
| C orig. | 0.50 | 0.50 | | | | | | | | |
| A test | 0.15 | 0.15 | 0.15 | 0.10 | 0.10 | 0.10 | 0.10 | 0.05 | 0.05 | 0.05 |
| B test | 0.30 | 0.25 | 0.20 | 0.15 | 0.10 | | | | | |
| C test | 0.60 | 0.40 | | | | | | | | |

other half of the job sets has 20 resources. Furthermore, in test sets VII and VIII all variations are allowed that were in test set VI.

### 5.2. Performance indicators of estimation quality

The estimation error ($e_i$), which is the difference between the estimated and the realized value of the interaction margin, can be used to evaluate the estimation quality of the models. The estimation error ($e_i$) is determined as follows:

$$e_i = I_i - \hat{I}_i. \tag{3}$$

The $R^2_{\text{est}}$ indicates how well the estimations fit the realized data, and is calculated as follows:

$$R^2_{\text{est}} = 1 - \frac{\sum (I_i - \hat{I}_i)^2}{\sum (I_i - \bar{I})^2}. \tag{4}$$

### 5.3. Results

#### 5.3.1. Interaction margin estimation quality of the regression models

Table 7 displays the interaction margin estimation quality expressed in the estimation $R^2$ of the five regression models (Table 3) at the eight different test sets.

#### 5.3.2. Interaction margin estimation quality of the neural networks

Table 8 displays the interaction margin estimation quality of the 10 NN-models on the eight different test sets expressed in $R^2_{\text{est}}$. The two most right columns display, for each test set, the average estimation quality and the standard deviation (S.D.) over the 10 models. Apart from test set eight, the estimation quality of the 10 different NN-models are robust for differences in random initializations and different learning-validation partitions (resulting in low standard deviations).

#### 5.3.3. Comparing the estimation quality of the regression model and the neural networks

To compare the interaction margin estimation quality of the regression model and the artificial neural networks Table 9 displays the estimation quality of the regression model and the average estimation quality over the 10 NNs. The last column contains the 95% confidence interval based on the one-sample $t$-test; an asterisk indicates a significantly better estimation result ($p < 0.05$) of the NN-models compared with the best regression model.

So far we have addressed the estimation of the interaction margin ($\hat{I}$). To support planners in the batch process industries to determine achievable job sets we use the estimate of the makespan ($\hat{C}_{\max}$):

$$\hat{C}_{\max} = (1 + \hat{I}) \cdot lb. \tag{5}$$

For test set VI we have estimated the makespan with regression model RM5 and the NNs. This test set has been chosen because it allows for many variations in job set characteristics. The average makespan of test set VI is 1250, with a standard deviation of 762. The makespan estimation results of both techniques are given in Table 10.

Table 7
Interaction margin estimation quality of the regression models

| Test set | $R^2_{\text{est}}$ | | | | |
|---|---|---|---|---|---|
| | RM1 | RM2 | RM3 | RM4 | RM5 |
| I | 0.83 | 0.86 | 0.88 | 0.88 | 0.88 |
| II | 0.85 | 0.87 | 0.89 | 0.88 | 0.89 |
| III | 0.77 | 0.84 | 0.86 | 0.86 | 0.86 |
| IV | 0.79 | 0.85 | 0.87 | 0.85 | 0.85 |
| V | 0.84 | 0.78 | 0.83 | 0.83 | 0.90 |
| VI | 0.73 | 0.74 | 0.81 | 0.81 | 0.87 |
| VII | 0.82 | 0.86 | 0.89 | 0.88 | 0.92 |
| VIII | 0.38 | 0.81 | 0.84 | 0.82 | 0.73 |
| Average | 0.75 | 0.83 | 0.86 | 0.85 | 0.86 |

Table 8
Estimation quality of the 10 NN-models on the eight different test sets expressed in $R^2_{est}$

| Test set | $R^2_{est}$ neural networks | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | NN1 | NN2 | NN3 | NN4 | NN5 | NN6 | NN7 | NN8 | NN9 | NN10 | Average | S.D. |
| I | 0.94 | 0.93 | 0.95 | 0.93 | 0.94 | 0.93 | 0.92 | 0.94 | 0.93 | 0.93 | 0.93 | 0.01 |
| II | 0.95 | 0.94 | 0.96 | 0.95 | 0.96 | 0.95 | 0.95 | 0.95 | 0.95 | 0.95 | 0.95 | 0.00 |
| III | 0.90 | 0.89 | 0.91 | 0.89 | 0.90 | 0.90 | 0.90 | 0.91 | 0.90 | 0.89 | 0.90 | 0.00 |
| IV | 0.88 | 0.85 | 0.86 | 0.87 | 0.89 | 0.88 | 0.84 | 0.87 | 0.83 | 0.87 | 0.86 | 0.02 |
| V | 0.93 | 0.93 | 0.93 | 0.93 | 0.93 | 0.93 | 0.94 | 0.93 | 0.92 | 0.93 | 0.93 | 0.01 |
| VI | 0.91 | 0.90 | 0.91 | 0.91 | 0.91 | 0.92 | 0.93 | 0.89 | 0.90 | 0.92 | 0.91 | 0.01 |
| VII | 0.91 | 0.85 | 0.92 | 0.89 | 0.92 | 0.94 | 0.93 | 0.91 | 0.91 | 0.95 | 0.91 | 0.02 |
| VIII | 0.70 | 0.73 | 0.65 | 0.70 | 0.68 | 0.74 | 0.71 | 0.65 | 0.64 | 0.70 | 0.69 | 0.04 |
| Average | 0.89 | 0.88 | 0.89 | 0.88 | 0.89 | 0.90 | 0.89 | 0.88 | 0.87 | 0.89 | | |

The two most right columns display the average estimation quality and the standard deviation (S.D.) over the 10 models.

Table 9
Comparison interaction margin estimation quality of the regression models and the artificial neural networks

| Test set | $R^2_{est}$ RM5 | Average $R^2_{est}$ NNs | 95% Confidence interval |
|---|---|---|---|
| I | 0.88 | 0.93 | 0.928–0.940* |
| II | 0.89 | 0.95 | 0.947–0.955* |
| III | 0.86 | 0.90 | 0.894–0.904* |
| IV | 0.85 | 0.86 | 0.850–0.878* |
| V | 0.90 | 0.93 | 0.927–0.933* |
| VI | 0.87 | 0.91 | 0.902–0.918* |
| VII | 0.92 | 0.91 | 0.893–0.933 |
| VIII | 0.73 | 0.69 | 0.665–0.715 |
| Average | 0.86 | 0.89 | |

Table 10
Comparison of the quality of the makespan estimate for test set VI of the regression model and the average over 10 neural networks models

| | RM5 | NN-models |
|---|---|---|
| SDE | 102 | 91 (±7.3) |
| $R^2_{est}$ | 0.982 | 0.986 (±0.002) |

The results indicate that both, the regression model and the neural network models, have a good makespan estimation quality (all $R^2_{est}$ are very close to 1.0). The average estimation result of the NN-models is slightly but significant ($p = 0.05$) better. Finally, the average standard deviation of the error (SDE) of the NN-models is significantly ($p = 0.05$) lower than the standard deviation of the regression model.

## 6. Discussion and conclusions

In this paper, we investigated whether it is possible to accurately estimate the makespan or interaction margin of a job set in multipurpose batch process industries. We focus on methods that estimate the makespan on simple characteristics of the jobs and the available resources. Five interaction variables are defined and an indication of their influence on the makespan is experimentally investigated. Two different techniques (i.e. regression analysis and neural network modelling) are examined to express a quantitative relationship between the five interaction variables of a job set and the amount of interaction for the regarding job set.

Using regression analysis we first developed simple models, which only include main effects,

and secondly more complex regression models, which include main effects and two-way interactions between variables. As a second modeling technique we used neural network models, which can handle complex non-linear relations. The advantage of the simple regression models is that they are easy to understand and the interaction between job set characteristics and job interaction becomes clear. Interpretation of models induced by neural networks is difficult: the trained neural networks do not provide any insight in the relation between the job set characteristics and the job interaction. Research is carried out that seeks for techniques to understand the knowledge stored in a trained neural network (Andrews et al., 1995; Weijters and van den Bosch, 1999).

On the basis of the presented experimental results we conclude that both regression models and neural network models give estimates of high quality. However, more complex regression models, and in an increasing degree, neural network models results in a higher estimation quality. The estimation quality of the neural network models appears significantly better than estimation quality of regression models. Tests showed that both types of estimation models are robust for changes in the job set characteristics. Only if there is a strong variation in total number of resources, the estimation quality of all estimation models decreases. Especially the neural network models appear sensitive for big variations in the total number of resources (estimation results on test set VIII). A more general explanation is the strong dependency of neural networks for similarity between training and new material.

The practical relevance of this research is that it is not necessary to construct a complete schedule for a given planning period to evaluate the achievability of a job set. Given some simple characteristics of the jobs and the available recourses we showed that both regression models and neural networks could realize good makespan estimates. Therefore, these models can support planners during capacity planning and order acceptance decisions. The estimation quality of both techniques is relatively robust, unless large changes in the resource configuration occur. In an industrial situation, the resource configuration generally remains unchanged for a relatively long period due to the high investments. Regression models with two-way interactions provide good estimation quality at the expense of insight into the relations. If planners in the multipurpose batch process industry prefer simple models that provide insight into the relation between job set characteristics and the achievability of a job set, then simple regression models are preferred. If insight into the relations is less important than the prediction quality, then neural network models are preferred.

For both, regression models and neural network models, historical data of job sets and the corresponding resulting makespan can be used for model induction. For both types of models the data set should be sufficiently large. If not enough historical data is available simulation can be used to generate data sets. This can be time consuming, but needs to be done only once. Given a sufficient amount of learning material the building of the estimation models is relatively simple and straightforward. Updating the models is only necessary when large changes occur in the resource configuration, because we have seen that in that case the estimation quality deteriorates considerably. If there is a strong dissimilarity between resource configuration characteristics of the training material and the actual job sets simple regression models deserve preference.

## References

Andrews, R., Diederich, J., Tickle, A.B., 1995. A survey and critique of techniques for extracting rules from trained artificial neural networks. Knowledge Based System 8, 373–389.

Arizono, I., Yamamoto, A., Ohta, H., 1992. Scheduling for minimizing total actual flow time by neural networks. International Journal of Production Research 30, 503–511.

Baker, K.R., Bertrand, J.W.M., 1981a. An investigation of due-date assignment rules with constrained tightness. Journal of Operations Management 1, 109–120.

Baker, K.R., Bertrand, J.W.M., 1981b. A comparison of due-date selection rules. IIE Transactions 13, 123–131.

Bertrand, J.W.M., 1983. The effects of workload dependent due-dates on job shop performance. Management Science 29, 799–816.

Chen, W., Muraki, M., 1997. An action strategy generation framework for on-line scheduling and control system in batch processing with neural networks. International Journal of Production Research 35, 3483–3506.

Cheng, T.C.E., Gupta, M.C., 1989. Survey of scheduling research involving due date determination decisions. European Journal of Operational Research 38, 156–166.

Eilon, S., Chowdhury, I.G., 1976. Due dates in job shop scheduling. International Journal of Production Research 14, 223–237.

Enns, S.T., 1993. Job shop flowtime prediction and tardiness control using queueing analysis. International Journal of Production Research 31, 2045–2057.

Enns, S.T., 1995. A dynamic forecasting model for job shop flowtime prediction and tardiness control. International Journal of Production Research 33, 1295–1312.

Enns, S.T., 1998. Lead time selection and behaviour of work flow in job shops. European Journal of Operational Research 109, 122–136.

Hill, T., Remus, W., 1994. Neural network models for intelligent support of managerial decision making. Decision Support Systems 11, 449–459.

Huang, C.L., Huang, Y.H., Chang, T.Y., Chang, S.H., Chung, C.H., Huang, D.T., Li, R.K., 1999. The construction of production performance prediction system for semiconductor manufacturing with artificial neural networks. International Journal of Production Research 37, 1387–1402.

Lawrence, S.R., 1995. Estimating flowtimes and setting due dates in complex production systems. IIE Transactions 27, 657–668.

Lee, Y.H., Kim, S., 1993. Neural network applications for scheduling jobs on parallel machines. Computer and Industrial Engineering 25, 227–230.

Lenstra, J.K., Rinnooy Kan, A.H.G., Brucker, P., 1977. Complexity of machine scheduling problems. Annals of Discrete Mathematics 1, 343–362.

Montgomery, D.C., Peck, E.A., 1992. Introduction to Linear Regression Analysis. Wiley, New York.

Philipoom, P.R., Rees, L.P., Wiegemann, L., 1994. Using neural networks to determine internally-set due-date assignments for shop scheduling. Decision Sciences 25, 825–851.

Prechelt, L., 1994. Proben1: A set of neural network benchmark problems and benchmarking rules. Technical Report 24/94, Fakultät für Informatik, Universität Karlsruhe, Germany.

Raaymakers, W.H.M., 1999. Order acceptance and capacity loading in batch process industries. Ph.D. Thesis, Eindhoven University of Technology.

Raaymakers, W.H.M., Bertrand, J.W.M., Fransoo, J.C., 2000. The performance of workload rules for order acceptance in batch chemical manufacturing. Journal of Intelligent Manufacturing 11, 217–228.

Raaymakers, W.H.M., Hoogeveen, J.A., 2000. Scheduling multipurpose batch process industries with no-wait restrictions by simulated annealing. European Journal of Operational Research 126, 131–151.

Raaymakers, W.H.M., Fransoo, J.C., 2000. Identification of aggregate resource and job set characteristics for predicting job set makespan in batch process industries. International Journal of Production Economics 68, 137–149.

Raaymakers, W.H.M., Bertrand, J.W.M., Fransoo, J.C., 2001. Makespan estimation in batch process industries using aggregate resource and job set characteristics. International Journal of Production Economics 70, 145–161.

Ragatz, G.L., Mabert, V.A., 1984. A simulation analysis of due date assignment rules. Journal of Operations Management 5, 27–39.

Rumelhart, D.E., Hinton, G.E., Williams, R.J., 1986. Learning internal representations by error propagation. In: Rumelhart, D.E., McClelland, J.L. (Eds.), Parallel Distributed Processing Explorations in the Microstructure of Cognition: Foundations, vol 1. MIT Press, Cambridge, MA, pp. 318–362.

Sabuncuoglu, I., Gurgun, B., 1996. A neural network model for scheduling problems. European Journal of Operational Reseach 93, 288–299.

Sabuncuoglu, I., 1998. Scheduling with neural networks – a review of the literature and new research directions. Production Planning and Control 9 (1), 2–12.

Statake, T., Katsumi, M., Nakamura, N., 1994. Neural network approach for minimizing the makespan of the general job–shop. International Journal of Production Economics 33, 67–74.

Vig, M.M., Dooley, K.J., 1991. Dynamic rules for due-date assignment. International Journal of Production Research 29, 1361–1377.

Vig, M.M., Dooley, K.J., 1993. Mixing static and dynamic flowtime estimates for due-date assignment. Journal of Operations Management 11, 67–79.

Weeks, J.K., 1979. A simulation study of predictable due-dates. Management Science 25, 363–373.

Weijters, A.J.M.M., van den Bosch, A.P.J., 1999. Interpreting knowledge representations in BP-SOM. Behaviormetrika 26, 107–129.