Dynamic scheduling of batch operations with non-identical machines
Zee, van der, D.J.; Harten, van, A.; Schuur, P.C.

Published: 01/01/1997

Document Version
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
• A submitted manuscript is the author's version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

Citation for published version (APA):

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.
• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal?

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 02. Jan. 2019
Dynamic Scheduling of Batch Operations
with Non-Identical Machines
D.J. van der Zee, A. van Harten,
P.C. Schuur

Preprints BETA: p-4
ISBN: 9036509424
ISSN:
Eindhoven / Twente, maart 1997

Keywords:

BETA-Research Programme: Information Tools
Dynamic Scheduling of Batch Operations with Non-Identical Machines

D.J. van der Zee*, A. van Harten and P.C. Schuur

Abstract- Batch-wise production is found in many industries. A good example of production systems which process products batch-wise are the ovens found in aircraft industry and in semiconductor manufacturing. These systems mostly consist of multiple machines of different types, given the range and volumes of products which have to be handled. Building on earlier research in aircraft industry, where the process of hardening synthetic aircraft parts was studied, we propose new strategies for the dynamic control of these type of systems: These so-called 'look-ahead strategies' base their decision to schedule a job on a certain machine upon the availability of information on a few near future arrivals. Moreover, a general method is presented, which shows how such strategies may be constructed. Through simulation the performance of the new control strategies was demonstrated in terms of logistical costs for several system configurations. Results of these simulations are compared with those for alternative strategies. Next to the information on relative performance of the control strategies also valuable new insights were obtained with regard to the relationship between system configuration and its performance. As an important special case the effect of machine grouping on system performance is studied.

1. Introduction

In a previous article (Van der Zee 1995) we introduced a new control strategy for the dynamic control of multi-server batch operations, the Dynamic Job Assignment Heuristic (DJAH). This so-called look-ahead strategy is characterized by the fact that it attempts to optimize system performance using its knowledge of just a few near future customer arrivals. In the aforementioned article we surveyed look-ahead strategies (Glassey 1991, 1993; Fowler 1992; Weng 1993; Robinson 1995) which address batch operations for identical machines. By an extensive series of simulations it has been shown that for most settings DJAH gives results which are at least as good or even better than existing look-ahead strategies. In contradiction to all other look-ahead strategies DJAH can be applied not only if minimization of average flow time is taken as a criterion for optimization, but also in case a more general minimization of logistic costs is considered. Moreover, while most look-ahead strategies only address the single machine case, DJAH also covers the case in which a system consists of multiple identical machines.

In industrial practice however, multi-server systems often consist of multiple machines of different types. The choice for different types of machines may be based on the required processing conditions (e.g. temperature, pressure), product characteristics (e.g. volume, dimensions) or operating costs (e.g. setup costs) or it is simply a matter of a historical growth pattern. The consideration of alternative machine types leads to a substantial increase in the complexity of the combinatorial problem faced by the planner. While for identical machines heuristic control was reduced to the scheduling of the currently available machine (Van der Zee 1995), such an approach seems less natural in case of alternative machine types. This would mean after all that the influence of machine characteristics on system performance is largely ignored. A 'better' machine can be available in the near future and it might be worthwhile to use this sort of information.

We discuss three new heuristic rules for controlling batch operations in this paper to deal with the case of non-identical machines. All three rules are based on the DJAH heuristic as defined in Van der Zee (1995). These rules originate from a general method. This method shows how a decision rule may be constructed using a multi-phase approach.

The main elements of the method are:

- Pre-selection of candidate machine/product type combinations.
- Sequential scanning of selected candidates.
- Actual dispatching of candidates resulting from the scanning procedure.

The latter two elements use iteration as an intrinsic ingredient. The way the iteration runs depends on whether one uses a pre-assignment of machines to product types or not.

* To whom correspondence should be addressed.
Benefits of the method are not only its guidance for model construction but also the improved understanding of the problem situation. For more details on the method see Section 2.

To gain insight in the potential of the new control strategies a simulation study has been carried out. Simulation experiments consider both systems which consist of identical and non-identical machines. Performance for the new control strategies is compared with that of an extended version of the Minimum Batch Size rule (Neuts 1967). Basically, the rule demands that a batch may start service once a certain minimum batch size is met. The essential difference between the Minimum Batch Size rule (MBS) and a look-ahead strategy lies in the fact that MBS makes no use of information on future arrivals. Furthermore the simulation study considers the concept of machine grouping for its impact on system performance. Machine grouping is much encountered in practice and involves the reduction of problem complexity by a fixed assignment of subsets of product types to machine groups. The question is whether this is profitable for the control strategies under consideration.

The contents of the paper are as follows: in Section 2 the construction is discussed of look-ahead strategies for the control of batch operations. It starts with a detailed description of system characteristics. Subsequently, the control problem is analyzed and a method is proposed for the construction of a suitable control strategy. In Section 3 we describe the new control strategies in more detail. Section 4 concerns an intermezzo which deals with two specific subjects we encountered during our research. It addresses the determination of the maximum possible throughput for a system consisting of non-identical machines. Such a quantity is e.g. important to establish the occupation rate or arrival intensity of a system. Besides, a rule for efficient machine grouping is discussed. The design of the simulation study is discussed in Section 5. Simulation results are presented in Section 6. Finally, conclusions are drawn in Section 7.

2. The construction of look-ahead strategies for batch operations

In this section we address the construction of new control strategies for batch operations, given the availability of multiple non-identical machines and a (very) limited knowledge of future product arrivals. Hereby we base ourselves on previous research, as reported in Van der Zee (1995, 1997). The question is how the DJAH heuristic, which addresses batch shops consisting of multiple identical machines, can be adapted to the case of non-identical machines. Therefore we first describe the shop environment. Secondly, the decision (scheduling) problem is formulated. In comparison with the case of multiple identical machines, the decision problem for the case of multiple non-identical machines appears to be much more complex. The additional complexity follows from the fact that an ‘efficient’ use of machinery requires a control strategy to consider the ‘fit’ of product and machine characteristics. While in the case of identical machines it makes no sense to consider alternative machines (as they have equal characteristics), one is here forced to do so. In order to formulate decision rules that can handle this complexity, we first define a general construction method. The method shows how a decision rule may be constructed by identifying a sequential (hierarchical) ordering of decisions. In Section 3 this method is used to adapt the DJAH heuristic for the case of non-identical machines.

First we shall describe the shop environment. Figure 1 serves to support our discussion. The picture graphically depicts a batch shop. Practical examples of such batch shops are the oven systems which can be found in aircraft industry and in semi-conductor manufacturing (Glassey 1991, Fowler 1992, Uzsoy 1992, Van der Zee 1995).

Next to a controller, the batch shop consists of a buffer and a number of servers (machines). The buffer is supposed to have an unlimited storage capacity. Machines process products batch-wise. It is demanded that batches are made up of products of the same type. This restriction follows logically from the fact that different types of products make different demands on processing conditions (e.g. temperature, pressure). Machines may be of different types. Differences between machine types are reflected by the required processing times or the allowed batch sizes (think of e.g. volume restrictions). In some cases the use of a particular machine can be restricted to a limited set of product types. Such restrictions may arise from e.g. the required processing conditions (pressure, temperature), product characteristics (size, dimensions) or cost considerations. The latter type of restrictions are not studied here. However, adapting the DJAH heuristic so as to include these restrictions is straightforward. The time needed for setup and the transportation of products between the buffer and a server is considered to be included in the processing time. The latter activities are sequence-independent, i.e., cost/time depends only on the next batch to be processed. The total amount of processing time needed to complete a job is fixed. While the processing time depends on machine and
product type, it is independent of the batch size. Pre-emption of a job, i.e., the interruption of processing, is not allowed as this would make products worthless for further use. We allow for the possibility of compound arrivals (c.f. Van der Zee 1996).

Figure 1 Batch shop

The above description defines the static, i.e., time independent, characteristics of the shop floor. Its dynamics is governed by two types of events:

- The actual arrival of products
- The completion of a machine job

Each of these events changes the shop status. As a consequence the controller may be forced to initiate some action (e.g. assign a new job to a machine that has become idle). In other words these events correspond to decision moments. It follows that, decisions are only taken if there is at least one machine available (in case of a product arrival) and there is at least one product in queue (in case a machine becomes available). Note that the reception of information on future product arrivals is not considered as a decision moment.

The above description of the shop floor sets the context for the decision problem. Let us now consider this problem in some more detail, making a distinction between an information base, decision options and a decision rule.

At a decision moment (indicated as $t_0$) a decision maker has to decide whether new jobs are initiated. His decision is based on information available on the shop status contained in the information base $\Gamma$. The information base is supposed to contain the following data:

- Queue length at $t_0$ for each product type $j$ ($q_j$).
- For each machine $m$, the moment $t_m^* \geq t_0$ when the machine is first available and if $t_m^* = t_0$ whether the machine is newly available or not (new, not new).
- For each product type $j$ the present and successive future arrivals $t_{k,j}$, ordered through the index $k$ according to moment of arrival, upto some specified information horizon.
- Lot size for each expected arrival ($L(t_{k,j})$).

Note that the first two data types concern information which is locally available, while the information on future arrivals is received from outside (e.g. other departments/companies). For look-ahead strategies it is assumed that the latter information only covers near future arrivals. For DJAH the set of expected arrivals $AR$ is defined as:
\[ AR = \{ t_{kj} | t_{kj} \leq t^\prime, j \in J, k = 1, 2, \ldots \}, \text{ with } t^\prime = \max_{m=1..M, j \in J} (t_{ij} + T_{mj}) \]  

(1)

In the above formula J denotes the set of product types. The look-ahead information horizon is represented by \( t^\prime \). It marks the moment in time until which information on future arrivals is assumed to be known. For DJAH a look-ahead horizon, i.e., information horizon, is chosen which relates to the first arrival of a product \( j \) and the required processing time on a machine \( m (T_{mj}) \). Many other choices for \( t^\prime \) are possible. However, we do not treat alternative choices here as they are dealt with elaborately in Van der Zee (1995), where a survey is presented of known look-ahead strategies including their different choices for the look-ahead horizon \( t^\prime \). It should be noted that information on future arrivals does not have to be complete or correct, i.e., forecasting errors or missing data are allowed, but they will of course influence the system’s performance. Related robustness questions will be dealt with briefly for this situation in the sequel.

The information contained in the information base \( \Gamma \) and the static characteristics of the batch shop (product and machine characteristics) determine the options which are open to the decision maker. A decision option refers to an assignment of quantities of available products to the machines available at \( t_0 \). An assignment \( S \) is defined as:

\[ S = \{ (m,j,d_{mj}) | 0 \leq d_{mj} \leq C_{mj}, j \in J, m = 1..M \} \]  

(2)

The decision variable \( d_{mj} \) indicates the number of products of type \( j \) to be loaded into machine \( m \) for processing at \( t_0 \). Of course the number of products loaded cannot exceed the machine capacity \( (C_{mj}) \), i.e., the maximum allowed batch size. Two constraints with regard to the choice of \( d_{mj} \) are:

\[ \begin{align*}
(i) & \quad d_{mj} = 0 \quad \text{if} \quad t^\prime_m > t_0 \\
(ii) & \quad \sum_{m=1..M} d_{mj} \leq q_j
\end{align*} \]  

(3)

The first restriction states that it is not possible to start a machine \( m \) that is not available at the decision moment. The second restriction says that the queue length \( (q_j) \) may not be exceeded. Note that the definition of the assignment \( S \) does not consider the possibility of scheduling a job at any other moment than the decision moment. In fact, with regard to a specific machine \( m \) (that is available at \( t_0 \)) there are two options:

1. The scheduling of a job at the decision moment (load, \( d_{mj} > 0 \)).
2. Postponement of the scheduling decision to the next decision moment (wait, \( d_{mj} = 0 \)).

Other look-ahead strategies, like the Minimum Cost Rate heuristic (Weng 1993), allow for the possibility of ‘jumping’. Jumping corresponds to the scheduling of a machine (that is available at \( t_0 \)) at the moment of some future product arrival. In this way it may skip (‘jump over’) some possible decision moments. A major disadvantage of such a policy is that information on future arrivals received in between, may be neglected. Especially, when forecast information on future arrivals is incomplete and/or less accurate, control heuristics which apply this policy show bad performance (c.f. Fowler 1992, Robinson 1995, Van der Zee 1995).

Another constraint with respect to the choice of \( d_{mj} \) in \( S \) was introduced in the beginning of this section. It concerns the requirement that batches should consist of one and the same type of product:

\[ \sum_{j \in J} a_{mj} \leq 1 \quad \text{with} \quad a_{mj} = \begin{cases} 
1 & \text{if } d_{mj} > 0 \\
0 & \text{otherwise}
\end{cases} \]  

(4)
I  Pre-selection of machine/product type combinations at \( t_0 \)

\[ \text{Result: } MP \in A \times P \]

II  Choice between no virtual pre-assignment (III) and virtual pre-assignment (III', IV')

III  Sequential scanning of pre-selected machine/product type combinations in case of no virtual pre-assignment

Iterative, per step:
(a) Scanning and reduction:
   1. Find next candidate(s) for scanning using a priority rule for the scanning order
      \[ \text{Result: } CMP \]
   2. Reduction to a unique candidate machine/product type combination
      \[ \text{Result: } m^*,j^* \]
(b) Dispatching: If the machine \( m^* \) is available, then decide between load/wait
      \[ \text{Result: } d_{m^*,i} \]

Next: update remaining candidates in MP and the information base for the next step
Repeat: until there are no candidates left

III'  Sequential scanning of pre-selected machine/product type combinations in case of virtual pre-assignment

Iterative, per step:
Select the next unique candidate machine/product type combination \( (m^*,j^*) \) according to a priority rule.
\[ \text{Result: } m^*,j^* \text{ (add to VAS)} \]
Next: update remaining candidates in MP and the information base for the next step.
Repeat: until no candidates are left
\[ \text{Result: } VAS = \{(m^*,j^*) | m^* \in A^v \subset A\} \]
\[ VAS_c \subseteq \{(m^*,j^*) \in VAS \mid m^* \in AV\} \]
And an inherent ordering on VAS and VAS_c

IV'  Dispatching

Iterative over VAS, according to its ordering:
Decide between load/wait for a pair \( (m^*,j^*) \in VAS \)
\[ \text{Result: } d_{m^*,j^*} \]
Next: update remaining elements in VAS and the information base for the next step.
Repeat: until there are no candidates left

Notation

\( j \) = Product type identifier
\( m \) = Machine identifier
\( J \) = The set of all product type identifiers
\( M \) = The number of machines
\( A \) = The set of machines to be considered for pre-selection of machine/product type combinations
\( AV \) = The set of machines available at \( t_0 \)
\( P \) = The set of product identifiers to be considered for pre-selection of machine/product type combinations
\( MP \) = The set of machine/product type combinations which results from pre-selection
\( CMP \) = The set of candidate machine/product type combinations which results from sequential scanning (III(a))
\( VAS \) = The set of virtual pre-assigned machine/product type combinations which results from sequential scanning (III')
\( VAS_c \) = Subset of VAS concerning only those machines that are available at \( t_0 \)
\( d_{m,j} \) = Decision variable indicating the number of products of type \( j \) which should be loaded into machine \( m \) at \( t_0 \).

Figure 2  A method for the construction of a decision rule \( d: J^v \rightarrow S \) at a decision moment \( t_0 \).
Here, the variable \( a_{m,j} \) indicates whether a batch of products of type \( j \) is loaded into machine \( m \) at \( t_o \). Further, \( d_{m,j} \) should be chosen in such a way that there is no 'waste' of capacity, i.e., if it is decided to load a machine \( m \), the batch size equals the minimum of queue length \( (q_j) \) and the used machine capacity:

\[
\sum_{m=1}^{M} d_{m,j} = \min(q_j, \sum_{m=1}^{M} a_{m,j} C_{m,j}) \ \forall \ j \in J
\]  

(5)

Note that this formula also allows that a product type \( j \) is loaded to more machines. If only one machine is available and it is decided to load a machine the formula reduces to \( d_{m,j} = \min(q_j, C_{m,j}) \).

Herewith we have established the structure of the input \((\Gamma)\) and the output \((S)\) of the process of decision making. Let us now consider the decision rule \( d \), which maps \( \Gamma \) on \( S \), given some decision moment \( t_o \):

\[
d: \Gamma \rightarrow S
\]  

(6)

In our introductory remarks of this section we already mentioned the inherent complexity of the decision problem for batch shops which consist of non-identical machines. The complexity mainly follows from the fact that if one wants to make good use of the different qualities of the available machines one is forced to consider alternative machines in the control strategy. This also includes alternative machines, which become available at a future moment. To deal with this combinatorial complexity we decided to devise a hierarchical decomposition method for the construction of decision rules. The construction of a rule should of course take the relevant performance criteria into account. As a criterion for optimization we adopted minimization of logistic costs. A problem is to find a suitable cost horizon over which such costs are computed. Besides, allocation of costs to different machines is not a trivial matter either. Logistic costs are assumed to be made up of waiting costs (e.g. relating to storage of products in the buffer) and setup costs (e.g. costs of manpower, energy and transportation). The minimization of average waiting time may be regarded as an important special case. Average waiting time is minimized in case waiting costs are linear and setup costs are absent. There is a close link between minimization of average waiting time and minimization of average flow time. For the case of a batch shop consisting of identical machines, minimization of average waiting time implies minimization of average flow time, given the fixed processing times. For the case of non-identical machines, however, there is a less straightforward relationship. We return to this issue in Section 6.

Let us now first outline our method in general terms, see Figure 2. In Section 3 this method will be used to adapt the DJAH heuristic for the case of non-identical machines. The method recognizes multiple phases for the construction of a decision rule. A survey of these phases is given in Figure 2.

Let us now discuss the method for the construction of decision rules in detail. Each of the phases for the construction of a decision rule as shown in Figure 2 leads to a procedure. Together these procedures make up the decision rule. In the sequel, the meaning of each of the construction phases is treated and examples are given of the possible logic behind rules which may be chosen for the construction of procedures. First an overview is given in Figure 3 of the numerous possibilities. We recognize that this overview is in no respect complete; we merely intend to confront the reader with some of the most interesting examples.

Let us now explain the ideas behind Figure 3 in more detail. In phase I of the method a procedure is defined for the pre-selection of machine/product type combinations (denoted as MP) which is allowed to be involved in decision making. This pre-selection may be based on any quantity directly derivable from \( C_{m,j}, T_{m,j} \) and the information base \( \Gamma \). 'Directly' means that no assumption is made on any specific pre-assignment of a certain product type to a certain machine, which would have an impact on the possibilities for further assignments. Such a 'sequential assignment' is the subject of phase III (or III, IV'). In this phase we typically pre-select on a basis of general exclusion principles. Exclusion, because it is a priori clear, that certain norms cannot be met. Hence, even in the best case (potentially), it is induced from \( \Gamma \), that some pre-specified upper or lower bound for some criterion will be violated. For example, it may be decided to neglect those
Pre-selection of machine, product type combinations at $t_0$

Restrictions on the set of product types ($P$)
- Queue length greater than zero

Restrictions on the set of machines ($A$)
- Only available machines
- The machine that has just completed its job (in case a machine becomes available)
- The machine that requires the shortest processing time (in case of a product arrival)

Restrictions on machine/product type combinations ($AxP$)
- Minimum potential fill rate of a machine for a product type
- Minimum potential throughput of a machine for a product type

Choice between no virtual pre-assignment (III) and virtual pre-assignment (III', IV')

Sequential scanning of pre-selected machine/product type combinations in case of no virtual assignment.

(a) Scanning and reduction
1. Select the set of candidate machine/product type combinations (CMP) using a scanning order: which machine/product types get priority
   Priority based on:
   - Machine identifier
   - Machine with maximum potential throughput
   - Moment a machine becomes available
   - Product identifier
   - Product type with largest share in product mix
   - Machine/product type combination with maximum potential throughput
2. Selection principles for unique assignment within CMP
   - Full loads as priority
   - Maximum potential throughput as priority
   - Total costs per unit as priority

(b) Dispatching: If $m^*$ is available, then the actual dispatching decision is based on a comparison of estimated costs for immediate processing ($m^*, j^*$) with alternatives

Structure elements:

Alternative starting moments
- First product arrival
- All product arrivals during fixed period

Alternative combinations
- no ($m^*, j^*$)
- yes (e.g. CMP)

Cost horizon
- The moment the next machine becomes available
- The moment the considered machine finishes the candidate job

Cost allocation
- Only waiting costs for product type $j^*$ which is considered plus a share in the waiting costs for those product types that are not considered for loading

Cost per unit
- Unit of processing time
- Unit of product

Sequential scanning of pre-selected machine/product type combinations in case of virtual assignment
See IIa1,2

Dispatching
See IIb

Figure 3 Examples of rules which may be adopted for the construction of subprocedures
combinations that will not guarantee a certain minimum fill rate for a machine on the basis of company standards. Essentially, limitations may be set on:

- The set of product types to be considered
- The set of machines to be considered
- Combinations of product types and machines

The set of product types to be considered is denoted as P. Note that P is a subset of J, i.e., the set of all product type identifiers. While \( P = J \) may be regarded as a default setting, other definitions are possible. For example, P may be limited to those product types for which the queue length is positive at \( t_0 \). This rule may be based on practical experience. Think e.g. of a situation where there is one machine and a large assortment of product types: the possibility that a product type for which the queue length is zero is loaded at a moment before the look-ahead horizon is not very likely. Similar restrictions can be put on the set of machines to be regarded (denoted as A). A logical choice for A in case of identical machines may be to focus on machines available at \( t_0 \) only. After all, considering alternative (busy) machines seems to make little sense as they have the same qualities. Even greedier choices are possible. For example, only consider the machine that has just completed its job (in case a machine becomes available) or only consider the machine which requires the shortest processing time for the type of products that have just arrived (in case of a product arrival). Finally, restrictions may also be set on machine/product type combinations (MP). In our introductory example the minimum fill rate has already been mentioned as a way to restrict the set of machine/product type combinations. A similar standard may be based on a certain minimum level for throughput. Both standards may arise from cost considerations: below a certain minimum level management regards costs per item as too high.

In phase II it is decided whether a choice is made for virtual pre-assignment or not. We will explain both approaches. The application of virtual pre-assignment supposes the reduction of MP to a set of machine product type combinations (VAS), in which each machine is assigned to one product type at the most. Such a 'scheme' is built up using a scanning procedure, which selects product machine combinations from MP using a priority rule. The idea behind the creation of such a scheme is that the final dispatching decision (to be discussed below) may be based on better cost estimates for starting up some machine, since the allocation of costs can be made in a more precise way. It is considered a 'virtual' pre-assignment because only available machines will be considered for dispatching, while the other (busy) machines are only included for a more accurate cost estimate.

While the above approach supposes a decoupling between scanning and dispatching, a logical alternative would be to couple both decisions. The coupling assumes an iterative process in which MP is scanned first for a suitable candidate machine product type combination. Once such a candidate is found a dispatching decision is immediately made. This process continues by calling on the scanning procedure again and ends by exhaustion of MP. Note that the difference between both approaches lies really in the knowledge of the pre-assignments of other machines.

Both approaches are worked out in detail in phases III, III' and IV' of the method. Phase III concerns the case of no virtual pre-assignment. Phases III', IV' address the case of virtual pre-assignment. The fact that we consider two additional phases in the latter case relates to the decoupling between scanning and dispatching procedures mentioned above.

Now we have come to phase III(a). While phase I addresses some rather coarse measures to reduce the allowed machine product type combinations, phase III is directed towards a more refined sequential scanning in view of reduction of MP. As a result of a scanning procedure, a reduced set of candidate machine/product type combinations (denoted as CMP) will be produced in each step. The difference with phase I is, that in this phase III, we install an iterative search for the machine/product type combinations (CMP) with the highest priority. The priority rule is an essential ingredient here, as well as the iteration. Given CMP, a unique element is selected and then tested for dispatching, the remaining elements of MP and the information base are updated. Hence, two subprocedures basically make up the scanning procedure:

a1: The establishment of a set of candidate machine/product type combinations CMP
a2: Reduction of candidates in CMP to a unique machine/product type combination
The first subprocedure (a1) is meant to realize those machine/product combinations which are expected to contribute most to the goal of cost minimization. To develop such a subprocedure one has to decide on some scanning order first, i.e., the priority rule has to be chosen by which elements of the set MP are considered. Note that the choice for a certain scanning order may strongly influence the outcome of the procedure. Think e.g. of a situation where in a step of the iteration, a product type is assigned to a machine becoming available at \( t' > t_0 \). Suppose that all combinations containing that product type are removed from the remainder of MP, because all available products can be handled. As a consequence a machine \( m \) in AV will then never be assigned to that product type. A scanning order priority rule can be:

- Machine related
- Product type related
- Related to both product and machine type

A simple example of a machine related scanning order is to focus on the machine identifier. More 'sophisticated' rules consider the potential throughput of a machine or the moment it becomes available. In a similar way, product related rules may be formulated. These rules may use the product identifier or the percentual share of a product type in the total product mix.

Given some rule for the scanning order, a supplementary priority rule should usually be defined for a unique selection of a candidate from the subset of machine/product type combinations (subprocedure (a2)). For example, consider a machine related scanning order. It may then be decided that if full loads are available for product types associated with a machine, priority is assigned to just these product types. A second example is prioritizing those product types which maximize the machine's potential throughput. Another possibility (which we will use in DJAH) is that the selection of a candidate already takes place in the sense of optimization of costs per unit. A unique assignment may very well follow automatically from a scanning order in which machine/product type combinations are scanned for maximum potential throughput.

In phase III(b) of our method, a procedure has to be defined which decides upon the actual dispatching of \((m^*, j^*) \in \text{CMP}\). If \( m^* \) corresponds with a machine available at \( t_0 \), dispatching involves a simple choice:

- Load/wait

Hence, it should be decided if the machine \( m^* \) is loaded at the decision moment (load, \( d_{m,j} > 0 \)) or if the decision is postponed to a future decision moment (wait, \( d_{m,j} = 0 \)). Estimated logistic costs are computed in case the associated job would be scheduled at the decision moment and compared with those found for \((m^*, j^*)\) or some alternatives starting at some later moment(s) in time. Typically, the latter moments correspond with expected product arrivals. For DJAH only the first arrival for each product type is considered. Other choices are possible, e.g. the Dynamic Batching Heuristic (Glassy 1991) regards all arrivals for a certain product type during a fixed period. Having established estimated costs for all options, the option is chosen which shows the minimum cost per unit in some sense. Costs may e.g. be computed per unit of processing time or per unit of product, both in recognition of the fact that it may be more efficient to load a large batch than a small one. If minimum costs are found for an option which requires waiting for a future arrival, a 'wait' decision is made, if not the corresponding machine is loaded with the product of the indicated type.

---

1 Note that the computation of the maximum potential throughput of a machine which is part of a system consisting of multiple non-identical machines is no small matter. In Subsection 4.1 we come back to this point.

2 The purpose of this priority rule is to prevent very long waiting times for product types which have relatively long processing times. In the example the selection process did not reduce the set of assignments to be considered (c.f. Fowler 1992).

3 Unless a full load is available at the decision moment and one decides \( d_{m,j} = C_{m,j} \).

4 Usually there is a close link between the optional starting moments that are considered by a control strategy and its assumed look-ahead horizon (c.f. Van der Zee 1995).
Above we characterized how look-ahead strategies attempt to solve the decision problems associated with dispatching by estimating logistic costs. The success of this approach strongly depends on the formulation of the underlying cost function. For this cost function we introduce $TV_{m,j}(t)$, with $t$ the moment a job is supposed to be scheduled for a machine/product type combination $(m,j)$, as follows:

$$TV_{m,j}(t) = \Phi_{m,j} + W_{m,j}^1(t) + \alpha_m W_{m,j}^2(t)$$  \hspace{1cm} (7)

In this definition $\Phi_{m,j}$ represents a fixed amount of setup costs. $W_{m,j}^1(t)$ denotes the estimated waiting costs up to a cost horizon $H_{m,j}(t)$ which are associated with the product type $j$ that is considered for loading into machine $m$ at time $t$. In a similar way $W_{m,j}^2(t)$ stands for the estimated waiting costs for product types other than $j$, given the proposed loading of products of type $j$ into $m$ at time $t$. In recognition of the fact that in systems consisting of multiple machines all these waiting costs do not necessarily have to be allocated to one and the same machine we introduce the parameter $\alpha_m$, denoting the percentage of $W_{m,j}^2(t)$ which should be associated with machine $m$. Let us now first discuss the definition of $W_{m,j}^1(t)$:

$$W_{m,j}^1(t_0) = \sum_{t_0 \leq t_0 < H_{m,j}} L(t_0) (H_{m,j} - t_0) \max(0, W_{m,j}(H_{m,j} - t_0) - q_0)$$

$$W_{m,j}^1(t_d) = \sum_{t_d \leq t_0 < H_{m,j}} L(t_0) (t_d - t_0) + \sum_{t_d \leq t_0 < H_{m,j}} L(t_0) (H_{m,j} - t_0) + \max(0, W_{m,j}(H_{m,j} - t_0) - q_j) \max(0, W_{m,j}(H_{m,j} - t_0) - q_j)$$  \hspace{1cm} (8)

We distinguished two formulas. The first formula addresses a situation where a machine $m$ is loaded with products of type $j$ at $t_0$, while the second formula addresses a similar situation where machine $m$ is started at the $d$-th arrival for products of type $j$ ($t_d$). For the associated cost horizons we write $H_{m,j}^0 = H(t_0)$ and $H_{m,j}^d = H(t_d)$. Although both formulas may be represented by a single formula, we prefer not to do so for the sake of clarity. The first formula starts by computing waiting costs for those products which cannot be included in the batch because the maximum batch size ($C_m$) would be exceeded. Then, costs are calculated for those products of type $j$ that arrive after $t_0$ up to the cost horizon $H_{m,j}^0$. In a similar way waiting costs are estimated for the situation in which a machine is assumed to start at $t_d$. First waiting costs are computed for the items of type $j$ that are in queue at $t_0$ ($q_j$). Then, just as in the previous situation, costs are calculated for those products of type $j$ that arrive before or after $t_d$ up to a cost horizon $H_{m,j}^d$. Note how the second formula is ‘corrected’ in the event that the sum of queue length and lot size of arriving products exceeds machine capacity. If this happens, additional costs are computed as indicated by the last term. Note that the cost horizon $H_{m,j}^0$ or $H_{m,j}^d$ in these definitions plays an essential role. Let us now consider the question of its definition. It seems a natural policy to associate a cost horizon with the moment another machine becomes available or the same machine becomes available again after having finished its job. For these moments offer new opportunities for loading. Hence the two main examples of cost horizons which relate to this idea are:

- The moment the next machine becomes available
- The moment the considered machine $m$ finishes the candidate job

Next to waiting costs for product type $j$ which is supposed to be loaded into machine $m$, waiting costs for other types of products have to be regarded up to a degree. We denoted these costs above as $\alpha_m W_{m,j}^2(t)$. Let us first discuss the definition of $W_{m,j}^2(t)$:

$$W_{m,j}^2(t_0) = \sum_{t_0 \leq t_0 < H_{m,j}} L(t_0) (H_{m,j} - t_0) \max(q_j - C_m, 0)$$

$$W_{m,j}^2(t_d) = \sum_{t_d \leq t_0 < H_{m,j}} L(t_0) (t_d - t_0) + \sum_{t_d \leq t_0 < H_{m,j}} L(t_0) (H_{m,j} - t_0) + \max(0, L(t_0) (t_d - t_0) + q_j - C_m, 0) \max(0, L(t_0) (t_d - t_0) + q_j - C_m, 0)$$  \hspace{1cm} (9)
Note the presence of the set $S_{pm}$ specifying the other relevant product types. Just like for $W_{ij}^k(t)$, two formulas are given, which are rather similar. The first term computes waiting costs for products of type $i$ that are in queue at $t_0(q)$, while the latter term represents waiting costs for products arriving up to the cost horizon. Not all product types (other than $j$) have to be included in $W_{ij}^k(t)$. Therefore, we select a subset $S_{pm}$ of $J$ based on some relevance criterion. Priorities which have already been introduced may be used as criteria for selection. Note how this relates the computation of waiting costs to the result of the scanning procedure. Besides, the allocation of the waiting costs can be controlled with the definition for $\alpha_m$. After all, if multiple machines are available, each machine may take its share of waiting costs. To estimate the share for a particular machine one may look e.g. to its percentual share in the total maximum possible throughput. Note how in Figure 3 the choices with respect to alternative starting moments, alternative combinations, cost horizon, cost allocation and the way costs per unit are defined, are mentioned as essential ingredients in phase III(b) of the method.

Finally we shall consider phases III' and IV', which are very similar to phases III(a) and III(b). Let us highlight the differences, the most important of which is the fact that the scanning procedure (III') and the dispatching procedure (IV') are decoupled.

In phase III' one may use the same logic as for phase III(a) in Figure 3 to build a set $VAS$. Above that, a set $VAS_0$ is also created. It contains only machine/product type combinations for which the machine is available at the decision moment, but $VAS_0$ does not necessarily has to concern all available machines. For example, $VAS_0$ may also be restricted to the machine that has just become available (in case a machine does become available) or to the machine that requires the shortest processing time (in case of a product arrival). The set $VAS_0$ is supposed to have an inherent ordering, which directly relates to the priority rules used for scanning.

Phase IV' is identical to III(b), except for the fact that the virtual pre-assignment (VAS) offers new possibilities for the definition of $S_{pm}$. For example, given VAS one may define $S_{pm}$ as the set of product types that are not considered for assignment to other machines. It is clear that these product types will have to wait.

Above the construction is treated of a decision rule for batch shops consisting of non-identical machines. Multiple construction phases are considered. Examples of possible approaches for constructing procedures have been given for each of the phases. Together these procedures should make up the decision rule. This method opens up a wealth of possible decision rules, altogether. In the next section we show how the method may be used to adapt the DJAH heuristic for the case of non-identical machines. Also a few extensions (DJAH1, DJAH2), taken on the basis of good intuition, will be considered there.

3. New control strategies for multi-server batch shops

In this section three new heuristic strategies are described for the control of batch shops which consist of multiple non-identical machines. All three control strategies relate to the DJAH heuristic as it has been defined for identical machines (Van der Zee 1995) and are developed using the method discussed in Section 2. Let us start by giving an overview of the main characteristics of each strategy (Figure 4).

In Figure 4 the three control strategies are characterized by the choices made with regard to their construction according to our general method. The three control strategies are referred to as DJAH, DJAH1, and DJAH2. As the first heuristic may be regarded as a straightforward extension of the DJAH heuristic developed for the case of multiple identical machines, it is also named DJAH. The other two heuristics share many of the characteristics of the original DJAH heuristic. On the other hand there are some differences as well, as will be shown in the sequel. Therefore they are referred to as DJAH1 and DJAH2. In the next two subsections (Subsections 3.1, 3.2) each of the control strategies is treated, using Figure 4 as a starting point for discussion. In Subsection 3.3 attention is paid to the Minimum Batch Size rule introduced by Neuts (1967). This rule will be used as a standard for comparison of simulation results for the new heuristic rules. Since it bases its scheduling decision on local information only, it can be used to show the added value of including information on future arrivals in decision making.

---

5 Note the relation between these rules and the ones mentioned in Phase I. See also Figure 3.
CONTROL STRATEGIES

PHASE

Pre-selection (I)

- Restrictions
  Machine(s) that become available (MB); Machine(s) available at \( t_0 \) that require(s) shortest processing time (PA)
  No restrictions
  No restrictions

Virtual pre-assignment (II)
  no
  yes
  yes

Sequential scanning (III(a)/III')

- Priority
  Moment a machine becomes available
  Machine/product type combination with maximum potential throughput
  Full loads as priority

- Selection principle
  Full loads as priority
  Cost per unit of product
  Full loads as priority
  Tie breaking rules
  Tie breaking rules

Dispatching (III(b)/IV')

- Alternative starting moments
  First product arrivals
  First product arrivals
  See DJAH,

- Alternative combinations
  yes
  no
  See DJAH,

- Cost horizon
  Next machine
  Considered machine
  See DJAH,

- Cost allocation
  All product types
  Product type which is loaded plus share in waiting costs of product types which are not considered for loading
  See DJAH,

- Cost per unit
  Unit of product
  Unit of product
  See DJAH,

Legend

MB = Decision moment corresponds with a machine which becomes available
PA = Decision moment corresponds with the arrival of products

Figure 4 An overview of three new control strategies

To guarantee a good understanding of the new look-ahead strategies introduced in this section, let us summarize the notation as it has been introduced in Section 2.

\[ j \] = Product type identifier
\[ m \] = Machine identifier (serial number)
\[ J \] = The set of product type identifiers \( j \)
\[ M \] = The number of machines
\[ N \] = The number of product types
\[ t_0 \] = The decision moment
\[ t_m \] = The moment after \( t_0 \) when machine \( m \) is available (again)
3.1 Dynamic Job Assignment Heuristic

The new DJAH heuristic is a straightforward extension of the DJAH heuristic introduced for batch shops consisting of identical machines (Van der Zee 1995). It should be mentioned that the extensions do not only consider its applicability to systems consisting of non-identical machines but also account for the possibility of compound arrivals. We will start by supplying the new algorithmic description of DJAH (10). Subsequently, its workings are explained in detail.

Pre-selection
At a decision moment the machine \( m \) that is to be considered for loading is selected first (see also Figure 4). In case the decision moment corresponds with the moment a machine becomes available, \( m \) is related to that machine. If more than one machine becomes available, the machine number ordering is decisive. Next, in case of a product arrival, the machine with the shortest processing time is selected among the machines available at \( t_k \). If more than one of the available machines require the same processing time, the machine with the smallest capacity greater than the queue length for the respective product type is chosen. If this still leads to a tie, the machine with the lowest number is chosen. Note that in the pre-selection we already focus on a unique machine. It should be remarked that DJAH sets no a priori restrictions to the set of product types involved in the decision. Hence, \( MP=\{m\} \times P \).

Virtual pre-assignment: no

Sequential scanning
Having established \( m \), it is considered if there are full loads for one or more product types \( (q_j \geq C_{m_j}) \). If full loads are available, the set of product types considered for dispatching is reduced to those product types for which a full load is available (compare the above formula). If on the other hand no full loads are available, all product types are considered candidates for dispatching. A unique selection for the product type, \( j^* \), is made by considering the logistic costs per unit of product.

Dispatching
The ‘best’ product type \( (j^*) \) for dispatching is now known. In order to decide whether it should be loaded, for each product type the logistic costs per unit of product are estimated also for the next possible starting moment. A starting moment corresponds with the moment a machine is assumed to be loaded. Hence we consider all alternatives in CMP. The starting moments correspond with the next arriving lot of products of a type \( j \) \((t_{kj}) \). These alternative starting moments are only considered in case there are no full loads present, of course. To estimate costs we adopted the cost function \( TV_{m_j}(t) \) as defined in Section 2.
Pre-selection of a machine $m$

If full loads are available

$$\text{then select product } j^* = \arg \min \frac{TV_{m_j}(t_0)}{C_{m_j}} \text{ and load machine } m$$

else select product $j^* = \arg \min_{j=1..N} \frac{1}{q_j} TV_{m_j}(t_0)$

$$\text{if } \frac{1}{q_j} TV_{m_j}(t_0) > \min_{j=1..N} \frac{1}{\min(q_j+L(t_{1,j}), C_{m_j})} TV_{m_j}(t_{1,j})$$

then wait
else load machine $m$

with $TV_{m_j}(t_0) = \Phi_{m_j} + (H_{m_j}^0 - t_0) \max(q_j - C_{m_j}, 0) + (H_{m_j}^0 - t_0) \sum_{i=1}^{N} q_i + \sum_{i=1}^{N} \Sigma_{k} L(t_{i,k}) (H_{m_j}^0 - t_{i,k})$

$$TV_{m_j}(t_{1,j}) = \Phi_{m_j} + q_j (t_{1,j} - t_0) + (H_{m_j}^1 - t_0) \sum_{i=1}^{N} q_i + \sum_{i=1}^{N} \Sigma_{k} L(t_{i,k}) (H_{m_j}^1 - t_{i,k}) + \max(0, L(t_{1,j}) + q_j - C_{m_j}) \max(H_{m_j}^1 - t_{1,j}, 0)$$

$$H_{m_j}^0 = \min_{a>m} (t_a, t_0 + T_{m_j})$$

$$H_{m_j}^1 = \min_{a>m} (t_a, t_{1,j} + T_{m_j})$$

(10)

considers setup costs ($\Phi_{m_j}$) and waiting costs. It requires the specification of a cost horizon ($H_{m_j}^d$), with $d = 0, 1$, and the way costs are allocated to the machines which have been distinguished ($\alpha_m, SP_m$). The moment the next machine becomes available has been chosen as a cost horizon. Note how the definition of $H_{m_j}^d$ recognizes the fact that in case of a relatively short processing time, $m$ itself may be the machine that becomes available first. Further, costs for all product types are considered, $\alpha_m = 1$, $SP_m = J$. Once costs have been estimated, they are weighed for the size of the batch before a final dispatching decision is made. This seems a natural policy from a business point of view, because it associates a 'cost price' with every item in the batch. If minimum costs are found for a situation in which the machine is loaded at $t_0$, the corresponding job is dispatched. If not, the next decision moment is waited for.

A last remark on the working of DJAH is about the fact that in some situations the iteration takes more than one step at a decision moment (i.e. CMP contains more than one machine). This situation may occur if for example a lot of products arrives, the size of which together with the queue length exceeds the capacity of the chosen machine $m$. If at the same time multiple machines are available at $t_0$, a first machine is considered and next the remaining machines.

The DJAH heuristic as described may be qualified as a greedy heuristic. The greediness arises mainly from its focus on one machine only (mentioned as $m$), i.e., usually it does not involve other machines in the decision. Especially in the case of systems consisting of non-identical machines, opportunities for performance improvement may well be neglected this way. For that reason, we developed two other control strategies that do consider alternative machine assignments, using virtual pre-assignment. These heuristics are treated in the next subsection.
3.2. New heuristics

In this subsection two new control strategies for batch operations are introduced. Given their similarity with the DJAH heuristic they are referred to as DJAH\textsubscript{1} and DJAH\textsubscript{2}. This similarity largely stems from the assumptions with respect to the look-ahead horizon and the decision options, as discussed in Section 2. The main difference is the choice for pre-assignment in the second phase. As can be concluded from Figure 4 both new control strategies show great similarity with respect to the construction of their decision rules. Therefore we do not treat them separately; only when it comes to the definition of the scanning procedure do we highlight their distinctive features.

Pre-selection

The new heuristics assume no a priori restrictions with regard to the machine/product type combinations to be involved in the decision.

Virtual pre-assignment: yes

Scanning

For the new heuristics a more elaborate scanning procedure has been developed, in which all machines are involved in the decision in principle. In line with our method, a distinction is made between scanning order and selection principle.

DJAH\textsubscript{1}

Priority

DJAH\textsubscript{1} uses a machine related scanning order which gives priority to a machine m\textsuperscript{*} which becomes available first. In case of a tie situation, i.e., two or more machines become available at the same moment, machines are chosen according to the highest potential throughput. In Subsection 4.1 it is shown how to compute this quantity. If this still leads to a tie, (e.g. in case of identical machines), a random selection is made.

Selection principle

Given the highest priority machine (according to the scanning order) DJAH\textsubscript{1}, first looks for full loads. If there are full loads only, those product types for which a full load is available are considered for further evaluation. Subsequently, one looks for the product type which would maximize throughput if the machine would be started at the moment it becomes available. Throughput (TH\textsubscript{m,j}) is defined as the quotient of batch size and processing time (T\textsubscript{m,j}):

\[
TH_{m,j} = \frac{\text{min}(\text{NA}(t_{m,j}'), C_{m,j})}{T_{m,j}}
\]  (11)

In this formula \text{NA}(t_{m,j}') denotes the sum of queue length and the arrivals for products of type j up to the moment a machine becomes available. In case there are multiple product types (denoted by the set MTR) for which the same maximum throughput is realized, the product j\textsuperscript{*} is chosen for which:

\[
j^* := \begin{cases} 
\text{argmin}_{j \in \text{MTR}} (t_{k,j} - t_{m,j}') & \text{if no full loads available} \\
\text{argmax}_{j \in \text{MTR}} \frac{\hat{q}_j C_{m,j}}{T_{m,j}} & \text{otherwise}
\end{cases}
\]  (12)

The formula states that if no full loads are available, the product type is selected for which the next arrival is the first to occur. In case of full loads, processing time and remaining queue length are considered to solve the tie situation. Admittedly, these are somewhat arbitrary rules. Their form is not a big issue, though, since they will rarely be used anyhow. Any other (for example stochastic) principle for a unique choice will also do the
trick with a hardly noticeable influence on the performance. Once $j^*$ is found, the combination $(m^*, j^*)$ is added to the set $VAS$ and a next machine is considered (following the scanning order). The data are updated by removing the load from $q_j$, $L(t_{kj})$ starting at $t_m$, next $t_{kj}$ etc. Usually, both machine and product type are no longer included in any further scanning operation. There is, however, one exception to this rule: if a full load has been assigned, the respective product type may sometimes be included in the scanning procedure. Note that a full load may be an indication of a long queue length and/or a high arrival rate. Scanning stops as soon as there are no machines or product types left to be considered.

The set of machines/product type combinations that are considered for dispatching ($VAS_0$) is found by selecting only those machine/product type combinations from $VAS$ which correspond to (depending on the fact which caused the decision moment):

- The machine(s) that has (have) become available
- The type of products that have just arrived

Note that usually one machine/product type pair is selected in $VAS_0$ at the most, but multiple pairs may be found. The latter situation may occur if e.g. multiple machines are available at $t_0$ and more than one of them is assigned to the same product type. They are treated according to the index numbers (see Section 2), found in the scanning order. On the other hand it may happen that the respective product type is assigned to a machine which is currently busy. Also it may occur that a machine that has become available is not assigned (e.g. in case the number of machines exceeds the number of product types). In those two cases no machine/product type pair is selected. Consequently, no dispatching decision is made.

**DJAH$_2$**

**priority**

DJAH$_2$ uses a scanning order which is machine and product type related. First the throughput is computed for all machines and all products types$^6$. As a definition of throughput we use:

$$TH_{m,j}^2 = \frac{\min(NA(t_m, T_{m,j}), C_{m,j})}{t_0 + T_{m,j} - t_m}$$  \hspace{1cm} (13)

If full loads for a machine are available, other combinations are ignored. Next we search for the combination with the best throughput. Note that the above definition for throughput differs slightly from the definition we presented earlier for DJAH$_1$. The denominator of the quotient accounts for the fact that machines become available at different moments. For DJAH$_1$ we do not have to bother about the moment a machine becomes available, because only one machine is evaluated at the same time.

**selection principle**

In case of tie situations, where the maximum throughput is realized for multiple machine/product type combinations, the tie-breaking rules mentioned in formula (12) are applied first. If this still leads to a tie because the same product type is considered, a random choice is made. The combination $(m^*, j^*)$ is added to $VAS$ and the data are updated as in DJAH$_1$. The reduction to $VAS_0$ is also the same as in DJAH$_1$.

**Dispatching** (for both DJAH$_1$ and DJAH$_2$)

For dispatching, the machine/product type combinations in $VAS_0$ are evaluated according to their index number. The dispatching decision is formulated as:

---

$^6$ Note that no restrictions were set in the first phase of our method: pre-selection. Compare also Figure 4.
if $q_j > C_{m_j}$
then load
else

if \( \frac{TV_{m_j}(t_0)}{q_j} > \frac{TV_{m_j}(t_1)}{\min(q_j, L(t_1), C_{m_j})} \)
then wait
else load

with $TV_{m_j}(t_0) = \Phi_{m_j} + \alpha_m((H_{m_j}^0 - t_0) \sum_{i \in SP_m} q_i + \sum_{i \in SP_m} \sum_{t_0 < t_k < t_0} L(t_k)(H_{m_j}^0 - t_k)) + \sum_{t_0 < t_k < t_0} L(t_k)(H_{m_j}^0 - t_k)$

$TV_{m_j}(t_1) = \Phi_{m_j} + q_j(t_1 - t_0) + \alpha_m((H_{m_j}^1 - t_0) \sum_{i \in SP_m} q_i + \sum_{i \in SP_m} \sum_{t_0 < t_k < h_{m_j}^1} L(t_k)(H_{m_j}^1 - t_k)) + \sum_{t_0 < t_k < h_{m_j}^1} L(t_k)(H_{m_j}^1 - t_k) + \max(0, L(t_1) + q_j - C_{m_j}) \max(H_{m_j}^1 - t_1, 0)$

$H_{m_j}^0 = t_0 + T_{m_j}$
$H_{m_j}^1 = t_1 + T_{m_j}$

The above formula states that if the queue length for product type $j$ exceeds the machine capacity, the machine is loaded right away. This seems a logical decision as it makes little sense to wait for more products to appear. If no full load is available, it is considered whether machine $m$ should be loaded right away or whether it is worthwhile to wait for the next arrival of products of type $j$. Note that only one product type is involved in the load/wait decision. In order to make this decision, the estimated logistic costs are computed for both options, using the cost function $TV_m(t)$ which has been introduced in Section 2. As a cost horizon $H_{m_j}^0$, with $d=0,1$, we choose the moment machine $m$ becomes available again. As regards cost allocation, we choose $\alpha_m$ equal to the fraction taken care of by machine $m$ in the maximum throughput of the total system. In Subsection 4.1 it is shown how to compute this fraction. We choose $SP_m = J \setminus VAS$, i.e., those product types which have not been considered in the scanning procedure. Let us illustrate this: imagine a situation with six product types and four machines. Application of the scanning procedure results in four machine/product type combinations with two product types left. In that case $SP_m$ consists of the two latter product types. Once costs have been estimated for both options, they are weighed for batch size (compare DJAH). Subsequently, the option is chosen for which minimum costs per unit of product are indicated. If this leads to the option in which loading of the machine is postponed to a next product arrival, the decision is to wait for a next decision moment, if not the machine is loaded.

3.3. Minimum Batch Size rule

The original MBS-rule addresses the single product type single machine case. It assumes that no information is available on future arrivals. As a consequence its criterion for optimization is reduced to an evaluation of the current situation only, which is determined by queue length. MBS compares queue length ($q$) with a fixed minimum batch size ($B$):

if $q \geq B$
then load the oven
else wait

(15)

Extensions of this rule have been proposed to make it fit for the multiple product types multiple machine case (Glassy 1993, Weng 1993). According to this rule, named MBSX, every time a machine cycle is completed, a new cycle is started right away provided that there are products in queue. Let us assume for a moment that a
unique machine becomes available. The type of product chosen is the one with the longest queue length. In case of a tie, the product which requires the shortest processing time, is loaded into the machine. If this still leads to a tie, then one makes a random choice. However, in case multiple machines are available, MBSX only covers the case of identical machines. It does not yet answer the question which type of machine to choose, if different types of machines are available at the decision moment. Also, what to do in the event that a large lot of products arrives in a similar setting, exceeding the maximum batch size for the chosen machine? For that we propose two extensions to MBSX. First an additional tie breaking rule is added. The rule states that in case multiple machines of different types are available the machine that requires the shortest processing time is chosen first. If this rule does not lead to a unique choice for a machine, then the one with the smallest capacity is chosen. If this does not resolve the tie, a random choice is made. To account for the possibility of compound arrivals, it is assumed that MBSX is applied iteratively with updated data in case the lot size of the arriving products exceeds the maximum batch size for the chosen machine.

4. Intermezzo on maximum possible throughput for batch shops and machine grouping

In this section two specific subjects are discussed which we encountered during our research. First the question is addressed how to determine the maximum possible throughput for a batch shop consisting of multiple non-identical machines (Subsection 4.1). The relevance of this question follows from the fact that without a notion of this quantity it is impossible to determine the occupation rate or traffic intensity of the system. Besides, the notion of maximum possible system throughput may be used in the construction of the decision rules for control strategies, as we have seen in Section 3. In Subsection 4.2 we deal with a situation often encountered in practical situations, where the complexity of the control problem is reduced by machine grouping. Machine grouping supposes a fixed assignment of a subset of product types to a subset of machines (machine group). A rule is considered which may help to find a cost efficient way for machine grouping.

4.1. Maximum system throughput

In Subsection 3.2 we encountered the problem of estimating the maximum possible throughput for a system with alternative machine types. This concept is defined as the maximum total number of products per time unit that can be processed by the system under the following conditions:

1. There is an abundant supply of product items to start a full load at any time.
2. The fraction of each product type \( j \) in the total product mix output by the system is pre-specified as \( s_j \).

The maximum possible throughput \( (F) \) can be determined by solving the following LP-problem:

\[
\begin{align*}
\text{max} & \quad F \\
\text{subject to:} & \\
F_j &= \sum_{m=1}^{M} C_{m,j} x_{m,j} \quad \forall j \in J \\
F_j &= s_j F \quad \forall j \in J \\
\sum_{j \in J} T_{m,j} x_{m,j} &\leq 1 \quad m = 1,2 \ldots M \\
F,F_j,x_{m,j} &\geq 0
\end{align*}
\]

In (16) the variables \( x_{m,j} \) are defined as the frequency with which product \( j \) is loaded into machine \( m \) in order to realize a maximum throughput \( (F) \) for the system. The system is supposed to operate at maximum capacity. In other words, only full loads \( (C_{m,j}) \) are supposed to be processed for each product type and for all machine types. This leads us to the first set of equalities for \( F_j \), the throughput of product type \( j \). The other relevant restrictions are:
The throughput $F_j$ should be the pre-specified fraction $s_j$ of the total throughput $F$ for each $j$.

The total time machine $m$ is occupied during a time period should not exceed the period length, given the required processing time ($T_{m,j}$) for each machine/product type combination for each $m$.

For example: Consider a system which consists of two machines ($m = 1, 2$), which handles two types of products ($J = \{1, 2\}$). Further specifications of the system are:

- $C_{11} = 2, T_{11} = 10, s_1 = 0.75$
- $C_{12} = 4, T_{12} = 25, s_2 = 0.25$
- $C_{21} = 4, T_{21} = 15$
- $C_{22} = 6, T_{22} = 20$

Application of an LP-solver gives as a solution:

- $x_{11} = 0.10; x_{12} = 0.00; x_{21} = 0.04; x_{22} = 0.02$
- $F_1 = 0.36; F_2 = 0.12$

Objective function value:

Total throughput $F = F_1 + F_2 = 0.48$ products per unit of time

Note how the solution to this problem also enables us to establish the corresponding throughput per machine: 0.2 for machine 1, 0.28 for machine 2.

4.2. Machine grouping

In practice, one often sees a tendency towards reduction of the complexity of the decision problem by setting limitations to the allowed combinations of machines and product types. This way, the planner makes the problem manageable for him- or herself. This seems a natural policy, especially in complex systems with machines that are more or less dedicated to specific product types this seems a natural policy. The first phase in our method is supposed to cover such a resource allocation. After this machine grouping, a decision problem of the same sort as before is found for each machine group and its corresponding product type group.

Let us discuss one particular way of reducing the decision problem. To this end, we partition the set $M_A$ of all machines into a collection of subsets $M_{Ar}$, $r = 1 .. R$, in such a way that the elements within each subset $M_A$, have more or less the same machine characteristics. Similarly, we construct a partitioning of the set $J$ of product types by grouping together those product types that have processing properties in common (product families). The next step is to allocate each subset $J_k$, $k = 1 .. K$, of product types to precisely one subset $M_{Ar}$ of machine types in such a way that the total operating cost ($OC_{r,k}$) per unit of time are minimized. In general these costs account for waiting and setup. These costs are supposed to be estimated by simulation. This can be done by solving the following assignment problem:

$$\min \sum_{r=1}^{R} \sum_{k=1}^{K} x_{r,k} OC_{r,k}$$

subject to:

$$\sum_{r=1}^{R} x_{r,k} = 1 \quad \forall \, k \in 1 .. K$$

$$\sum_{k=1}^{K} x_{r,k} = 1 \quad \forall \, r \in 1 .. R$$

with

$$x_{r,k} = \begin{cases} 1 & \text{if } M_{Ar} \text{ is assigned to } J_k \\ 0 & \text{else} \end{cases}$$

By reducing the problem in the above manner, a simple method is obtained to generate near optimal solutions to the original decision problem. Moreover, assuming the simulation results are accurate, we arrive at an
estimated upper bound of the original objective value. In Section 6 we will discuss the influence on system performance of a split up of a system into smaller job shops.

We suppose that all machines are suitable for processing all types of products. This might not be the case of course in many practical situations, given the specific requirements for certain product types which can not be met by all machines. However, it is straightforward to adapt the formulation of the assignment problem to include these restrictions.

5. Design of the simulation study

A number of simulations was carried out for different system configurations in order to gain insight in systems performance, given the application of the new heuristics (DJAH, DJAH₁, DJAH₂). In this section the design of the simulation study is discussed.

<table>
<thead>
<tr>
<th>Factor</th>
<th>I</th>
<th>II</th>
<th>III</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Criterion</td>
<td>Minimum average flow time; Minimum cost (Cₜ=60)</td>
<td>Minimum average flow time</td>
<td>Minimum average flow time</td>
</tr>
<tr>
<td>2. Control Strategy</td>
<td>MBSX, DJAH, DJAH₁, DJAH₂</td>
<td>MBSX, DJAH, DJAH₁, DJAH₂</td>
<td>DJAH</td>
</tr>
<tr>
<td>3. Interarrival Distribution</td>
<td>Exponential, Uniform</td>
<td>Exponential</td>
<td>Exponential</td>
</tr>
<tr>
<td>4. Quality of Information</td>
<td>Known, Predicted, Missing Data</td>
<td>Known</td>
<td>Known</td>
</tr>
<tr>
<td>5. No Machines/ No Products</td>
<td>4/2 2/4</td>
<td>4/2 2/4</td>
<td>2/1 1/2</td>
</tr>
<tr>
<td>7. Capacity per product (Cᵢ)</td>
<td>(5,5), (7,3), (5,5,5,5), (8,6,4,2)</td>
<td>(5,5,5,5), (7,3,3), (5,5), (7,3)</td>
<td>(5,5), (7,3), 5, (7,3)</td>
</tr>
<tr>
<td>8. Capacity per machine (Cₘ)</td>
<td>(25,25), (40,10), (25,25,25,25), (40,30,20,10)</td>
<td>(25,25,25,25), (20,33%,33%), (25,25), (20,33%))</td>
<td>(25,25), (20,33%), 25, (20:33%)</td>
</tr>
<tr>
<td>9. Processing Time for product (Tᵢ)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10. Processing Time for machine (Tₘ)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11. Traffic Intensity (p)</td>
<td>0.3, 0.6, 0.9</td>
<td>0.3, 0.6, 0.9</td>
<td>0.3, 0.6, 0.9</td>
</tr>
<tr>
<td>Number of Simulations</td>
<td>150</td>
<td>24</td>
<td>9</td>
</tr>
</tbody>
</table>

Table 1 Design of the Simulation Study

The simulations concern experimental situations, which were chosen in line with those used in previous experiments (Van der Zee 1995, 1996). This way a comparison of simulation results is facilitated. Three series of experiments were carried out. In Table 1 an overview is given of these series. The first series (I)
regards identical machines. This is interesting, because a comparison with previous results (see Van der Zee 1995) will provide us with an impression of the quality of our new heuristics compared with the ones studied previously. System configurations which consist of machines of alternative types are considered in a second series of simulations (II). The third series of simulations (III) is supplemental to both other series. It considers the effects of a fixed assignment of machine groups to product families (see also Section 4.2). For all simulations the lot size of the arriving products is assumed to be one. Note that the influence of compound arrivals on system performance was studied in Van der Zee (1996).

In the first series of simulations we test the performance of MBSX, DJAH, DJAH \textsuperscript{1}, DJAH \textsuperscript{2} for two experimental situations. A distinction is made between an example situation in which the number of machines is dominant (N=2, M=4) and a configuration for which the number of product types is dominant (N=4, M=2). The criteria for optimization which were adopted, were the minimum average flow time criterion and the minimum cost criterion. Logistic costs are supposed to consist of waiting costs and setup costs. Waiting costs equal 1 per unit of time, while for setup costs a fixed amount of 60 was assumed.

For each of the experimental situations a default setting has been defined for product and machine characteristics. This setting is marked in bold in Table 1 (factors 3-10). Alternative system configurations are chosen by changing the value for exactly one of the configuration variables (factors). Note that for factors 5-10, an explicit distinction is made between the settings for both situations. In cases of identical machines it is clear that $C_{m,j} = C_j$, $T_{m,j} = T_j$ for all $m$. Their values are found under factors 7 and 9. In cases of non-identical machines we assumed for the sake of simplicity, that $C_{m,j}$ is product independent, i.e., $= C_m$ and $T_{m,j}$ is machine independent, i.e., $= T_j$.

The interarrival time distributions we used are the Poisson distribution and a uniform distribution. The Poisson distribution is fully defined by the mean arrival rate $\lambda$. The range for the uniform distribution was set as $[0.5/\lambda, 1.5/\lambda]$, i.e., here also $1/\lambda$ is the mean interarrival time. Given a value for the traffic intensity ($\rho$) and the knowledge of the maximum possible throughput of the system (mpts), the exact value for $\lambda$ can be computed from the following expression (cf. Chaudry (1983)):

$$\rho = \frac{\lambda}{\text{mpts}}$$

In Subsection 4.1 we have shown how mpts can be determined using an LP-formulation. Because work load tends to have a major impact on the performance of a queueing system, all settings mentioned were analyzed for low (30%), moderate (60%) and high (90%) traffic intensities.

In robustness tests for the heuristics, we consider the influence of forecasting errors and incomplete data (factor 4). These tests are included because such situations occur frequently in practice. Forecasting errors are assumed to be normally distributed with mean equal to zero and a standard deviation which equals half the standard deviation of the interarrival times ($\sigma$). In the simulation model forecasting errors are associated with the data the decision maker receives on future arrival moments. Note that the possibility of forecasting errors requires a more refined updating of the information set on future arrivals (AR). At each decision moment corresponding with a product arrival the forecasted arrival moment for this product is removed from AR. Further, for decision making arrivals that are forecasted at $t$, but in ‘reality’ occur at a later than $t$, are ignored once the decision moment $t$ passes $t$.

Also performance for heuristics in situations in which the decision maker lacks on average 50% of the data on future arrivals is tested. These situations are modeled by associating a chance of 0.5 with each arriving product that it is not reported to the decision maker before its actual arrival. As a consequence the heuristics have to base their decision on the knowledge of later arrivals.

Similarly, a number of simulations (II) has been carried out for system configurations which consist of multiple machines of alternative types. Because the influence on system performance of most decision variables (factors) has already been considered in the first series of simulations, the second series of simulations focusses on machine characteristics. Machine characteristics considered are the required machine dependent processing times ($T_m$) and the product independent maximum batch size allowed ($C_m$). Note that the default settings have already been covered by the previous series of simulations.

Next to the above mentioned series of simulations, we carried out a small supplementary series of simulations (III). These simulations are needed to consider the effects of a fixed assignment of a subset of machines to a product family on system performance.
The package which was used to carry out the simulation experiments is ExSpect (Bakkenist 1994). ExSpect is a Petri Nets-based analysis tool. It allows for structural analysis as well as dynamic analysis by simulation. To facilitate the modeling process a logistics reference model was adopted (Zee van der 1997). A simulation model built according to this reference model can easily be adapted to incorporate new control rules or even new control structures. Moreover, the principles of object oriented design (see e.g. Booch, 1994) underlying both ExSpect and the reference model, guarantee reusability of model components in order to support further research.

With regard to the reliability of our experiments, similar remarks apply as in Van der Zee (1995). In particular, in case average waiting (flow) time is taken as a criterion for optimization, the standard deviation of the average waiting (flow) time for traffic intensities of 30 and 60% is in the order of 0-0.2% of the average waiting (flow) time. For traffic intensities of 90% standard deviation is in the order of 1.3% of the average waiting (flow) time. In case minimum cost is taken as a criterion, the respective values are 0-0.1% (traffic intensities of 30-60%) and 1.0% (traffic intensities of 90%).

6. Analysis of simulation results

In Section 5 the design of a simulation study has been discussed. In this section we address the results for these simulations. Three series of simulation experiments will be distinguished in line with Section 5. First, simulation results are considered for the case of identical machines, in Subsection 6.1. In the following subsection, attention will be paid to systems which consist of multiple alternative machine types. Finally, in Subsection 6.3, a few simulation results are treated, which illustrate the impact of a fixed allocation of machines on system performance. It should be noticed that settings for the simulation experiments will be indicated by either default or non-default factor which is studied, as mentioned in Section 5 (Table 1). For settings which consider the influence of product-related characteristics on system performance, we will only briefly comment on simulation results. These type of settings have been studied elaborately in previous work (e.g. Fowler 1992, Glassey 1993 and Van der Zee 1995, 1996).

6.1. Identical machines

In the Tables 2a,b an overview is given of simulation results for an experimental situation in which two types of products (N=2) are processed on four identical machines (M=4).

<table>
<thead>
<tr>
<th>No</th>
<th>Factor</th>
<th>p</th>
<th>MBSX</th>
<th>DJAH</th>
<th>DJAH</th>
<th>DJAH</th>
<th>Δ₁</th>
<th>Δ₂</th>
<th>Δ₃</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>default</td>
<td>0.3</td>
<td>4.68</td>
<td>2.67</td>
<td>3.81</td>
<td>5.03</td>
<td>42.9</td>
<td>18.6</td>
<td>-7.5</td>
</tr>
<tr>
<td>2</td>
<td>default</td>
<td>0.6</td>
<td>6.21</td>
<td>4.21</td>
<td>4.36</td>
<td>5.26</td>
<td>32.2</td>
<td>29.8</td>
<td>15.3</td>
</tr>
<tr>
<td>3</td>
<td>default</td>
<td>0.9</td>
<td>10.68</td>
<td>9.02</td>
<td>8.92</td>
<td>9.04</td>
<td>16.9</td>
<td>17.9</td>
<td>15.4</td>
</tr>
</tbody>
</table>

\[ \Delta₁ = 100\% \left( \frac{MBSX - DJAH}{MBSX} \right) \]
\[ \Delta₂ = 100\% \left( \frac{DJAH_1 - DJAH}{DJAH} \right) \]
\[ \Delta₃ = 100\% \left( \frac{MBSX - DJAH_2}{MBSX} \right) \]

Table 2a Average waiting time for N = 2, M = 4 (identical machines)

<table>
<thead>
<tr>
<th>No</th>
<th>Factor</th>
<th>p</th>
<th>MBSX</th>
<th>DJAH</th>
<th>DJAH</th>
<th>DJAH</th>
<th>Δ₁</th>
<th>Δ₂</th>
<th>Δ₃</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>default</td>
<td>0.3</td>
<td>42.35</td>
<td>24.05</td>
<td>23.77</td>
<td>23.67</td>
<td>43.2</td>
<td>43.9</td>
<td>44.1</td>
</tr>
<tr>
<td>2</td>
<td>default</td>
<td>0.6</td>
<td>26.16</td>
<td>19.23</td>
<td>19.37</td>
<td>19.21</td>
<td>26.5</td>
<td>26.0</td>
<td>26.6</td>
</tr>
<tr>
<td>3</td>
<td>default</td>
<td>0.9</td>
<td>24.17</td>
<td>21.25</td>
<td>21.35</td>
<td>21.38</td>
<td>12.1</td>
<td>12.7</td>
<td>11.5</td>
</tr>
</tbody>
</table>

\[ \Delta₁ = 100\% \left( \frac{MBSX - DJAH}{MBSX} \right) \]
\[ \Delta₂ = 100\% \left( \frac{MBSX - DJAH}{MBSX} \right) \]

Table 2b Average cost price per unit of product for N = 2, M = 4 (identical machines)

The performance of MBSX, DJAH, DJAH₁ and DJAH₂ are indicated for each of the three default settings with increasing p. The third column gives the traffic intensity (p). The last three columns represent percentual differences between MBSX and the three look-ahead strategies. A positive difference indicates that the look-ahead strategy in question performs better. The heuristics are tested according to two performance criteria: average waiting time and average cost price. In the latter case operational costs are not only composed of waiting costs but also of setup costs. Note that for identical machines the average waiting time criterion is
identical to the average flow time criterion (see also Section 2). Simulation results are presented with an accuracy of two decimals. The accuracy is determined by the design of the simulation study (Section 5). To test the statistical validity of the differences a paired t-approach has been used with a 95% confidence interval (see e.g. Law (1991)).

The results in the Tables 2a,b clearly indicate the improvement of system performance which is obtained if a look-ahead strategy is used instead of MBSX, which bases its decision on local information only. This conclusion is valid for both the average flow (waiting) time criterion and the average cost price criterion, except for the case in which traffic intensity is low and DJAH₂ is applied. We will return to this point in a moment. In general, improvements are large for low traffic intensities and smaller for high traffic intensities. This is due to the saturation effect, which was described by Glassey et al. (1993). They state that in case of high traffic intensities:

(1) A large fraction of the time, decision options are limited to the loading of full batches. As a consequence the ‘look ahead’ heuristics and the ‘greedy rule’, i.e., MBSX, more often take the same decision.
(2) The larger the queue, the less likely it is that the total waiting time will be reduced by delaying the start until the next arrival.

As a consequence the improvement of look-ahead strategies over the MBSX rule diminishes. As can be concluded from Tables 2a,b the inclusion of setup costs has a ‘smoothing effect’ on the relative performance of the look-ahead strategies for moderate and high traffic intensities.

If one compares performances for DJAH, DJAH₁ and DJAH₂ it is apparent that differences are relatively small if the average cost price per unit of product is taken as a criterion for optimization. This is probably due to the fact that the fixed setup costs already make up a large part of the cost price, leaving less room for percentual improvement. However, if the average waiting time is taken as a criterion for optimization, significant differences in performance are observed. For low and moderate traffic intensities, DJAH outperforms DJAH₁ and DJAH₂. It is remarkable that if DJAH₂ is applied for low traffic intensities, system performance is even worse than if MBSX is taken as a control strategy. We consider two reasons which may cause this behavior:

(1) The choice of a cost horizon
(2) The greediness coming from the throughput orientation in phase III’

In Section 2 we stated that it seems natural to associate a cost horizon with a moment a machine becomes available. Remember that for both DJAH₁ and DJAH₂ a cost horizon is chosen which corresponds to the moment ‘the considered machine’ finishes its job. However, in case there are a large number of machines, such a choice may sometimes be less fortunate, because a suitable machine may become available earlier. As a consequence, our estimate of logistic costs which supports the dispatching decision, may become less accurate. This in turn leads to a worse system performance. Another reason for the relatively bad performance may arise from the throughput orientation which characterizes the scanning procedure. It is observed that often a machine is chosen which becomes available later on, rather than a machine that is currently idle. Estimated throughput for the first machine is then higher than for the latter machine. This is caused mainly by the higher value for NA_j(t_m) in the computation of throughput, i.e., by the number of arrivals up to the moment machine m becomes available (see Subsection 3.2). Whereas the first machine may be confronted with only a few items in queue (NA_j(t_m) is small), the second machine can include all incoming arrivals up to the moment it becomes available. As is to be expected this effect will be stronger for low and moderate traffic intensities. After all, the weight of queue length in the computation of NA_j(t_m) increases strongly for high traffic intensities.

Let us now consider the effects of alternative settings for each of the factors considered in Section 5, Table 1. To constrain our simulation efforts, only DJAH and MBSX are considered. We chose DJAH instead of the other two look-ahead strategies because of its performance. The results for these settings are shown in Table 3. Note that the first three rows concern the default settings. Alternative settings are indicated in the second column.
Table 3  Average waiting time / cost price per unit of product for $N = 2$, $M = 4$ (alternative settings)

<table>
<thead>
<tr>
<th>No</th>
<th>Factor</th>
<th>$\rho$</th>
<th>MBSX</th>
<th>DJAH</th>
<th>$\Delta_i$</th>
<th>MBSX</th>
<th>DJAH</th>
<th>$\Delta_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>default</td>
<td>0.3</td>
<td>4.68</td>
<td>2.67</td>
<td>42.9</td>
<td>42.35</td>
<td>24.05</td>
<td>43.2</td>
</tr>
<tr>
<td>2</td>
<td>default</td>
<td>0.6</td>
<td>6.21</td>
<td>4.21</td>
<td>32.2</td>
<td>26.16</td>
<td>19.23</td>
<td>26.5</td>
</tr>
<tr>
<td>3</td>
<td>default</td>
<td>0.9</td>
<td>10.66</td>
<td>9.02</td>
<td>16.9</td>
<td>24.17</td>
<td>21.25</td>
<td>12.1</td>
</tr>
<tr>
<td>4</td>
<td>uniform arrivals</td>
<td>0.3</td>
<td>4.83</td>
<td>2.72</td>
<td>43.7</td>
<td>44.85</td>
<td>25.35</td>
<td>43.5</td>
</tr>
<tr>
<td>5</td>
<td>uniform arrivals</td>
<td>0.6</td>
<td>8.53</td>
<td>5.24</td>
<td>50.3</td>
<td>28.53</td>
<td>19.59</td>
<td>31.3</td>
</tr>
<tr>
<td>6</td>
<td>uniform arrivals</td>
<td>0.9</td>
<td>12.15</td>
<td>5.54</td>
<td>54.4</td>
<td>25.49</td>
<td>17.87</td>
<td>29.9</td>
</tr>
<tr>
<td>7</td>
<td>product mix</td>
<td>0.3</td>
<td>4.37</td>
<td>2.41</td>
<td>44.9</td>
<td>41.83</td>
<td>23.48</td>
<td>44.0</td>
</tr>
<tr>
<td>8</td>
<td>product mix</td>
<td>0.6</td>
<td>5.98</td>
<td>3.98</td>
<td>33.4</td>
<td>25.92</td>
<td>18.97</td>
<td>26.8</td>
</tr>
<tr>
<td>9</td>
<td>product mix</td>
<td>0.9</td>
<td>10.86</td>
<td>8.91</td>
<td>18.0</td>
<td>24.17</td>
<td>21.13</td>
<td>12.6</td>
</tr>
<tr>
<td>10</td>
<td>capacity</td>
<td>0.3</td>
<td>3.75</td>
<td>2.12</td>
<td>43.5</td>
<td>49.17</td>
<td>27.77</td>
<td>43.5</td>
</tr>
<tr>
<td>11</td>
<td>capacity</td>
<td>0.6</td>
<td>6.40</td>
<td>4.10</td>
<td>36.0</td>
<td>32.22</td>
<td>22.83</td>
<td>29.1</td>
</tr>
<tr>
<td>12</td>
<td>capacity</td>
<td>0.9</td>
<td>10.26</td>
<td>7.61</td>
<td>25.8</td>
<td>27.59</td>
<td>22.78</td>
<td>17.4</td>
</tr>
<tr>
<td>13</td>
<td>service time</td>
<td>0.3</td>
<td>5.23</td>
<td>2.92</td>
<td>44.2</td>
<td>42.42</td>
<td>23.97</td>
<td>43.5</td>
</tr>
<tr>
<td>14</td>
<td>service time</td>
<td>0.6</td>
<td>7.30</td>
<td>4.69</td>
<td>35.8</td>
<td>27.30</td>
<td>19.27</td>
<td>29.4</td>
</tr>
<tr>
<td>15</td>
<td>service time</td>
<td>0.9</td>
<td>14.63</td>
<td>10.74</td>
<td>26.6</td>
<td>27.95</td>
<td>22.19</td>
<td>20.6</td>
</tr>
<tr>
<td>16</td>
<td>data missing</td>
<td>0.3</td>
<td>4.68</td>
<td>3.49</td>
<td>25.4</td>
<td>42.35</td>
<td>28.03</td>
<td>33.8</td>
</tr>
<tr>
<td>17</td>
<td>data missing</td>
<td>0.6</td>
<td>6.21</td>
<td>5.02</td>
<td>16.1</td>
<td>26.16</td>
<td>20.96</td>
<td>19.9</td>
</tr>
<tr>
<td>18</td>
<td>data missing</td>
<td>0.9</td>
<td>10.86</td>
<td>9.83</td>
<td>9.5</td>
<td>24.17</td>
<td>22.05</td>
<td>8.8</td>
</tr>
<tr>
<td>19</td>
<td>data forecast</td>
<td>0.3</td>
<td>4.68</td>
<td>3.42</td>
<td>26.9</td>
<td>42.35</td>
<td>25.24</td>
<td>40.4</td>
</tr>
<tr>
<td>20</td>
<td>data forecast</td>
<td>0.6</td>
<td>6.21</td>
<td>4.72</td>
<td>24.0</td>
<td>26.16</td>
<td>19.66</td>
<td>24.8</td>
</tr>
<tr>
<td>21</td>
<td>data forecast</td>
<td>0.9</td>
<td>10.86</td>
<td>9.31</td>
<td>14.3</td>
<td>24.17</td>
<td>21.39</td>
<td>11.5</td>
</tr>
</tbody>
</table>

$\Delta_i = 100^\ast(MBSX - DJAH)/MBSX$

In case arrivals are known (settings 1-15), the relative performance of DJAH (ninth column) is less for moderate and high traffic intensities in comparison with similar settings for which setup costs are absent (sixth column). On the other hand, if the information on future arrivals is uncertain (settings 16-21), the relative performance is less influenced.

An exception to the 'rule' that the relative performance of a look-ahead strategy diminishes in comparison with MBSX for higher traffic intensities, is found for settings 4-6. Results for these settings show increasing values for the relative performance of DJAH in case average waiting time is adopted as a criterion. This is probably due to the fact that MBSX can be considered as a greedy version of the MBS heuristic. It just chooses the product type with the longest queue length, without considering the balancing of machine use. Machine use can be better balanced by setting a minimum batch size. This would be in line with the original MBS heuristic. However, it is very difficult to find such a minimum batch size (see Van der Zee 1995). As can be concluded from Table 2 the 'punishment' for this greedy policy is more severe for uniform arrivals than for Poisson arrivals (compare settings 1-3 and 4-6). In other words, the relative performance of DJAH is influenced by the regularity of the arrival pattern.

A change of the product mix by attributing different shares to the product types results in lower values for the average waiting time and the average cost price. Since this 'reduces' the variety in products to be handled a reduction of waiting costs is what is to be expected (c.f. Van der Zee (1996)).

In the settings 10-12 we consider the effects of product dependent capacities on system performance. It should be remarked that a direct comparison with the default settings is not possible, because the maximum service rate for settings 10-12 is lower than the maximum service rate for the default settings (see also Subsection 4.1 and Section 5). As a consequence, the arrival rate, i.e., the number of products arriving per unit of time, is lower for each setting of traffic intensity. This leads to a lower work load for the system, i.e., the time averaged number of products in process in the system decreases. Glassey (1993) found that lower loads lead to an increase of relative performance of look-ahead strategies in comparison with MBS. This experience is confirmed by our experiments.

Performance in case of product dependent service times (settings 13-15) is clearly influenced by the long processing times which are required for some product types. Although the other product types require shorter processing times this does not make up for the performance loss.

The robustness tests (settings 16-21) indicate the influence the quality of information has on system performance. Although the influence on relative performance of DJAH is great, the relative performance of DJAH improvement over MBSX is still significant.
Let us now consider the other experimental situation, in which four product types have to be processed on two identical machines (c.f. Van der Zee 1995). In Table 4a,b simulation results are shown for MBSX, DJAH, DJAH₁, and DJAH₂.

<table>
<thead>
<tr>
<th>No</th>
<th>Factor</th>
<th>p</th>
<th>MBSX</th>
<th>DJAH</th>
<th>DJAH₁</th>
<th>DJAH₂</th>
<th>Δ₁</th>
<th>Δ₂</th>
<th>Δ₃</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>default</td>
<td>0.3</td>
<td>15.20</td>
<td>12.11</td>
<td>10.98</td>
<td>10.73</td>
<td>20.3</td>
<td>27.8</td>
<td>29.4</td>
</tr>
<tr>
<td>2</td>
<td>default</td>
<td>0.6</td>
<td>20.95</td>
<td>18.16</td>
<td>17.74</td>
<td>17.49</td>
<td>13.3</td>
<td>15.3</td>
<td>16.5</td>
</tr>
<tr>
<td>3</td>
<td>default</td>
<td>0.9</td>
<td>32.00</td>
<td>29.44</td>
<td>29.32</td>
<td>29.25</td>
<td>8.0</td>
<td>8.4</td>
<td>8.6</td>
</tr>
</tbody>
</table>

\[
\Delta₁ = 100\times \frac{(MBSX - DJAH)}{MBSX} \\
\Delta₂ = 100\times \frac{(MBSX - DJAH₁)}{MBSX} \\
\Delta₃ = 100\times \frac{(MBSX - DJAH₂)}{MBSX}
\]

Table 4a Average waiting time for N = 4, M = 2 (identical machines)

<table>
<thead>
<tr>
<th>No</th>
<th>Factor</th>
<th>p</th>
<th>MBSX</th>
<th>DJAH</th>
<th>DJAH₁</th>
<th>DJAH₂</th>
<th>Δ₁</th>
<th>Δ₂</th>
<th>Δ₃</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>default</td>
<td>0.3</td>
<td>53.43</td>
<td>41.81</td>
<td>40.23</td>
<td>40.42</td>
<td>21.7</td>
<td>24.7</td>
<td>24.4</td>
</tr>
<tr>
<td>2</td>
<td>default</td>
<td>0.6</td>
<td>40.92</td>
<td>35.62</td>
<td>35.15</td>
<td>35.28</td>
<td>13.0</td>
<td>14.1</td>
<td>13.8</td>
</tr>
<tr>
<td>3</td>
<td>default</td>
<td>0.9</td>
<td>45.33</td>
<td>42.73</td>
<td>42.11</td>
<td>42.39</td>
<td>5.7</td>
<td>7.1</td>
<td>6.5</td>
</tr>
</tbody>
</table>

Table 4b Average cost price per unit of product for N = 4, M = 2 (identical machines)

If the results in Tables 4a,b are compared with those in Tables 2a,b, it appears that the relative performance for DJAH (and DJAH₁, DJAH₂) has worsened. This is caused by the fact that the ratio of products and machines has been altered. As a consequence of the larger product assortment, less profit is to be gained by postponing the loading of the machine because other products have to wait.

Tables 4a,b indicate that for most settings DJAH₂ yields the best results. The improvements of system performance if DJAH₂ is applied instead of DJAH are significant. In case the average flow time is taken as a criterion, improvements in the order of 10% are found for low traffic intensities, while for moderate traffic intensities improvements still amount to about 5%. As one would expect, given the saturation effect described above, improvements for high traffic intensities are small (about 1%). If optimization is directed towards the minimization of the average cost price per unit of product, improvements are somewhat less, but still significant. It should be remarked, however, that the performance for DJAH₁ is close or equal to that for DJAH₂. One of the reasons of the improvements of DJAH₁, DJAH₂ over DJAH is probably the better ‘fit’ of the cost horizon. Above we found that this fit may be worse for situations in which there is a relatively large number of machines (in comparison with the number of product types). However, in this case a relatively large number of product types is considered. In such situations there will often be no opportunity to start a new batch of a certain product type, until the moment the currently available machine(s) finishes its/their job(s). Therefore the choice of a cost horizon as in DJAH₁, DJAH₂ seems more adequate in this kind of situation. Besides, in situations where the number of product types is dominant over the number of machines, it seems even more important to make an efficient use of machinery. After all, the multitude of product types strongly worsens system performance. Therefore the greedy throughput orientation of DJAH₁, DJAH₂ seems a good approach. What is more, given the fact that in general larger queue lengths are realized for higher product dominated settings, this effect will relatively be strengthened for those settings.

Just as in the first situation, we studied the effects of product characteristics and quality of information on future arrivals on systems performance for MBSX and the best look-ahead strategy. In our experience, DJAH₂ is the best strategy. The simulation results for both strategies are shown in Table 5. The results are in line with the results presented for the first situation. The main difference lies in the fact that the relative performance of the look-ahead strategy DJAH₁ in comparison with that of MBSX is somewhat less. As explained above, this is caused by the multitude of product types which reduces the opportunities for postponement of a decision. As a consequence, there are fewer possibilities to improve on system performance (c.f. Van der Zee 1995).
Dynamic Scheduling of Batch Operations with Non-Identical Machines

6.2. Non-identical machines

In the previous subsection, two different experimental situations have been studied which consisted of a number of identical machines. While the first example considers a situation in which the number of machines is dominant, the second example addresses settings in which the number of products is dominant. In this subsection, simulation results for these situations will be compared to similar example situations which consist of a number of machines of different types. The type of a machine is supposed to be determined by its capacity, i.e., the allowed batch sizes, and the required processing times for each product type. System performance is measured according to the average flow time criterion. Note that in case of alternative machine types, minimization of the average flow time does not automatically correspond with minimization of the average waiting time. After all, the choice for 'faster' machines influences flow times. For that reason some tables will present values for average flow time instead of values for the average waiting time.

The first example concerns a system which consists of four machines, which process two types of products. The machines can be classified in two types: two large machines with a maximum batch size of seven products and two small machines which allow for a batch size of three products for each of the product types. Except for these characteristics, all other system characteristics are equal to the default settings adopted for the case of identical machines (see the first example in Subsection 6.1). In the same way an alternative setting is studied, where a distinction is made in two fast and two slow machine types: fast machines require a processing time of 20 time units for each product type, while slow machines require 33% time units, but all machines have the same capacity 5. Simulation results for these experiments are presented in the Tables 6 and 7. Note that in Table 7 results concern average flow times, whereas Table 6 refers to average waiting times. In order to facilitate comparison of results both tables also present average flow (waiting) times for the identical machines case (in brackets).

<table>
<thead>
<tr>
<th>No</th>
<th>Factor</th>
<th>p</th>
<th>MBSX</th>
<th>DJAH</th>
<th>Δ₁</th>
<th>MBSX</th>
<th>DJAH</th>
<th>Δ₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>default</td>
<td>0.3</td>
<td>15.20</td>
<td>10.73</td>
<td>29.4</td>
<td>53.43</td>
<td>40.42</td>
<td>24.4</td>
</tr>
<tr>
<td>2</td>
<td>default</td>
<td>0.6</td>
<td>20.95</td>
<td>17.49</td>
<td>16.5</td>
<td>40.92</td>
<td>35.28</td>
<td>13.8</td>
</tr>
<tr>
<td>3</td>
<td>default</td>
<td>0.9</td>
<td>32.00</td>
<td>29.25</td>
<td>8.6</td>
<td>45.33</td>
<td>42.39</td>
<td>6.5</td>
</tr>
<tr>
<td>4</td>
<td>uniform arrivals</td>
<td>0.3</td>
<td>15.69</td>
<td>10.66</td>
<td>32.1</td>
<td>39.88</td>
<td>35.68</td>
<td>10.5</td>
</tr>
<tr>
<td>5</td>
<td>uniform arrivals</td>
<td>0.6</td>
<td>21.16</td>
<td>16.85</td>
<td>20.4</td>
<td>41.16</td>
<td>34.94</td>
<td>15.1</td>
</tr>
<tr>
<td>6</td>
<td>uniform arrivals</td>
<td>0.9</td>
<td>26.54</td>
<td>22.79</td>
<td>14.1</td>
<td>45.33</td>
<td>42.39</td>
<td>12.6</td>
</tr>
<tr>
<td>7</td>
<td>product mix</td>
<td>0.3</td>
<td>13.26</td>
<td>9.00</td>
<td>32.1</td>
<td>51.17</td>
<td>37.36</td>
<td>27.0</td>
</tr>
<tr>
<td>8</td>
<td>product mix</td>
<td>0.6</td>
<td>19.41</td>
<td>15.96</td>
<td>17.8</td>
<td>39.38</td>
<td>33.40</td>
<td>15.2</td>
</tr>
<tr>
<td>9</td>
<td>product mix</td>
<td>0.9</td>
<td>30.96</td>
<td>28.51</td>
<td>7.9</td>
<td>44.28</td>
<td>41.32</td>
<td>6.7</td>
</tr>
<tr>
<td>10</td>
<td>capacity</td>
<td>0.3</td>
<td>10.35</td>
<td>7.28</td>
<td>29.7</td>
<td>59.57</td>
<td>44.10</td>
<td>26.0</td>
</tr>
<tr>
<td>11</td>
<td>capacity</td>
<td>0.6</td>
<td>23.38</td>
<td>18.33</td>
<td>21.6</td>
<td>49.37</td>
<td>41.23</td>
<td>16.5</td>
</tr>
<tr>
<td>12</td>
<td>capacity</td>
<td>0.9</td>
<td>47.97</td>
<td>34.90</td>
<td>27.2</td>
<td>65.33</td>
<td>51.69</td>
<td>20.9</td>
</tr>
<tr>
<td>13</td>
<td>service time</td>
<td>0.3</td>
<td>14.62</td>
<td>9.87</td>
<td>32.5</td>
<td>53.86</td>
<td>40.71</td>
<td>24.4</td>
</tr>
<tr>
<td>14</td>
<td>service time</td>
<td>0.6</td>
<td>20.88</td>
<td>17.17</td>
<td>17.8</td>
<td>41.41</td>
<td>36.18</td>
<td>12.6</td>
</tr>
<tr>
<td>15</td>
<td>service time</td>
<td>0.9</td>
<td>33.99</td>
<td>30.02</td>
<td>11.7</td>
<td>47.40</td>
<td>43.28</td>
<td>8.7</td>
</tr>
<tr>
<td>16</td>
<td>data missing</td>
<td>0.3</td>
<td>15.20</td>
<td>12.53</td>
<td>17.6</td>
<td>53.43</td>
<td>42.43</td>
<td>20.6</td>
</tr>
<tr>
<td>17</td>
<td>data missing</td>
<td>0.6</td>
<td>19.41</td>
<td>15.96</td>
<td>12.3</td>
<td>39.38</td>
<td>33.40</td>
<td>15.2</td>
</tr>
<tr>
<td>18</td>
<td>data missing</td>
<td>0.9</td>
<td>30.96</td>
<td>28.51</td>
<td>7.9</td>
<td>44.28</td>
<td>41.32</td>
<td>6.7</td>
</tr>
<tr>
<td>19</td>
<td>data forecast</td>
<td>0.3</td>
<td>15.20</td>
<td>12.53</td>
<td>17.6</td>
<td>53.43</td>
<td>42.43</td>
<td>20.6</td>
</tr>
<tr>
<td>20</td>
<td>data forecast</td>
<td>0.6</td>
<td>19.41</td>
<td>15.96</td>
<td>12.3</td>
<td>39.38</td>
<td>33.40</td>
<td>15.2</td>
</tr>
<tr>
<td>21</td>
<td>data forecast</td>
<td>0.9</td>
<td>30.96</td>
<td>28.51</td>
<td>7.9</td>
<td>44.28</td>
<td>41.32</td>
<td>6.7</td>
</tr>
</tbody>
</table>

Δ₁ = 100*(MBSX - DJAH)/MBSX

Table 5 Average waiting time / cost price per unit of product for N = 4, M = 2 (alternative settings)

The first example concerns a system which consists of four machines, which process two types of products. The machines can be classified in two types: two large machines with a maximum batch size of seven products and two small machines which allow for a batch size of three products for each of the product types. Except for these characteristics, all other system characteristics are equal to the default settings adopted for the case of identical machines (see the first example in Subsection 6.1). In the same way an alternative setting is studied, where a distinction is made in two fast and two slow machine types: fast machines require a processing time of 20 time units for each product type, while slow machines require 33% time units, but all machines have the same capacity 5. Simulation results for these experiments are presented in the Tables 6 and 7. Note that in Table 7 results concern average flow times, whereas Table 6 refers to average waiting times. In order to facilitate comparison of results both tables also present average flow (waiting) times for the identical machines case (in brackets).

<table>
<thead>
<tr>
<th>No</th>
<th>Factor</th>
<th>p</th>
<th>MBSX</th>
<th>DJAH</th>
<th>Δ₁</th>
<th>MBSX</th>
<th>DJAH</th>
<th>Δ₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>capacity</td>
<td>0.3</td>
<td>5.36</td>
<td>4.50</td>
<td>52.5</td>
<td>47.5</td>
<td>4.95(4.68)</td>
<td>2.90</td>
</tr>
<tr>
<td>2</td>
<td>capacity</td>
<td>0.6</td>
<td>7.83</td>
<td>6.97</td>
<td>43.1</td>
<td>56.9</td>
<td>7.34(6.21)</td>
<td>4.78</td>
</tr>
<tr>
<td>3</td>
<td>capacity</td>
<td>0.9</td>
<td>13.45</td>
<td>12.76</td>
<td>32.9</td>
<td>67.1</td>
<td>12.99(10.86)</td>
<td>8.84</td>
</tr>
</tbody>
</table>

Δ₁ = 100*(MBSX - DJAH)/MBSX

wᵢ = waiting time per product for machines of type i
%ᵢ = percentage of arrivals handled by machines of type i
avg = average waiting (flow) time

Table 6 Average waiting time for N = 2, M = 4 (alternative machine types)
The results in Table 6 suggest that the choice for machines with different capacities instead of machines with identical capacities worsens system performance. On the other hand, if one considers the situation in which processing times differ per machine (Table 7), it is found that system performance is not much influenced (in comparison with the case of identical machines). Note that the influence of machine capacities on system performance is stronger for higher traffic intensities. This is as expected, since only then capacities become restrictive.

Simulations are carried out for a situation similar to the second example defined in Subsection 6.1 in order to gain insight in system performance in cases where the number of product types is dominant over the number of machines. Whereas in the first example of this subsection we supposed two machines of each type to be available, here only one machine of each type is considered. Simulation results are presented in Tables 8 and 9. Just like in Subsection 6.1 results for DJAH 2 are compared with those of MBSX.

The results in Tables 8 and 9 are quite similar to those in Tables 6 and 7. It is illustrative to see how DJAH 2 tries to optimize performance. This is clearly expressed by the values for the percentage of arrivals handled by a machine i (%) in Table 7: In contrast with MBSX, DJAH 2 follows a throughput-oriented policy which tries to take advantage of the availability of faster machines and it succeeds.

Finally, we discuss the influence of a fixed allocation of machine groups to product groups on system performance, as suggested in Subsection 4.2. First we address the machine dominant situations (four machines, two product types) which were described earlier in Subsection 6.1 and 6.2. Fixed allocation is realized by splitting up the original batch shop in two identical smaller shops of two machines each.

### Table 7 Average flow time for N = 2, M = 4 (alternative machine types)

<table>
<thead>
<tr>
<th>No</th>
<th>Factor</th>
<th>ρ</th>
<th>MBSX</th>
<th>DJAH</th>
<th>Δi</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>f1</td>
<td>f2</td>
<td>σ1</td>
<td>σ2</td>
</tr>
<tr>
<td>1</td>
<td>serv.time</td>
<td>0.3</td>
<td>24.91</td>
<td>37.94</td>
<td>63.8</td>
</tr>
<tr>
<td>2</td>
<td>serv.time</td>
<td>0.6</td>
<td>26.40</td>
<td>40.01</td>
<td>63.1</td>
</tr>
<tr>
<td>3</td>
<td>serv.time</td>
<td>0.9</td>
<td>31.27</td>
<td>45.00</td>
<td>62.5</td>
</tr>
</tbody>
</table>

\[ \Delta_i = \frac{100}{\text{MBSX}} \times (\text{MBSX} - \text{MBSX}) \]

\[ f_i = \text{flow time per product for machines of type } i \]

\[ %_i = \text{percentage of arrivals handled by machines of type } i \]

\[ \text{avg} = \text{average waiting (flow) time} \]

### Table 8 Average waiting time for N = 4, M = 2 (alternative machine types)

<table>
<thead>
<tr>
<th>No</th>
<th>Factor</th>
<th>ρ</th>
<th>MBSX</th>
<th>DJAH</th>
<th>Δi</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>w1</td>
<td>w2</td>
<td>σ1</td>
<td>σ2</td>
</tr>
<tr>
<td>1</td>
<td>capacity</td>
<td>0.3</td>
<td>15.80</td>
<td>15.19</td>
<td>51.6</td>
</tr>
<tr>
<td>2</td>
<td>capacity</td>
<td>0.6</td>
<td>27.70</td>
<td>23.11</td>
<td>45.4</td>
</tr>
<tr>
<td>3</td>
<td>capacity</td>
<td>0.9</td>
<td>46.93</td>
<td>35.84</td>
<td>33.1</td>
</tr>
</tbody>
</table>

\[ \Delta_i = \frac{100}{\text{MBSX}} \times (\text{MBSX} - \text{MBSX}) \]

\[ w_i = \text{waiting time per product for machines of type } i \]

\[ %_i = \text{percentage of arrivals handled by machines of type } i \]

\[ \text{avg} = \text{average waiting (flow) time} \]

### Table 9 Average flow time for N = 4, M = 2 (alternative machine types)

<table>
<thead>
<tr>
<th>No</th>
<th>Factor</th>
<th>ρ</th>
<th>MBSX</th>
<th>DJAH</th>
<th>Δi</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>f1</td>
<td>f2</td>
<td>σ1</td>
<td>σ2</td>
</tr>
<tr>
<td>1</td>
<td>serv.time</td>
<td>0.3</td>
<td>35.14</td>
<td>48.92</td>
<td>63.1</td>
</tr>
<tr>
<td>2</td>
<td>serv.time</td>
<td>0.6</td>
<td>41.03</td>
<td>54.78</td>
<td>62.9</td>
</tr>
<tr>
<td>3</td>
<td>serv.time</td>
<td>0.9</td>
<td>51.75</td>
<td>65.99</td>
<td>62.1</td>
</tr>
</tbody>
</table>

\[ \Delta_i = \frac{100}{\text{MBSX}} \times (\text{MBSX} - \text{MBSX}) \]

\[ f_i = \text{flow time per product for machines of type } i \]

\[ %_i = \text{percentage of arrivals handled by machines of type } i \]

\[ \text{avg} = \text{average waiting (flow) time} \]

6.3. **Machine grouping**

Finally, we discuss the influence of a fixed allocation of machine groups to product groups on system performance, as suggested in Subsection 4.2. First we address the machine dominant situations (four machines, two product types) which were described earlier in Subsection 6.1 and 6.2. Fixed allocation is realized by splitting up the original batch shop in two identical smaller shops of two machines each.
(MA₁={1,2}; MA₂={3,4})⁷. Besides, we considered all possible partitions of the set of product types J. Note that only two such partitions are possible: J₁={1,2} and J₂={1,2}. Subsequently, given some partitioning, the assignment procedure mentioned in Subsection 4.2 is applied. Because the subsets MA₁, MA₂ concern identical shops (involving the same operating costs) such an assignment is easily found. For example, consider the case in which J is partitioned in J₁={1}, J₂={2}. A possible assignment would be x₁,₁; x₂,₂=1. Next, in a similar way, the product dominant situations are addressed (two machines, four product types). Here the two machine batch shop is split up into two smaller shops of one machine each. Note that both shops are not necessarily identical (since we also consider non-identical machines). In order to find an allocation of machinery for which operational costs are minimal, we use our findings in previous research (Van der Zee 1995, 1996). In the latter article we studied the influence of the number of product types and the number of machines on system performance.

Let us first discuss the case in which the four machine batch shop is split up in two two-machine shops. In Table 10 an overview is given of a number of simulations. Results are shown for DJAH both for the case in which the system is regarded as a single job shop (S₁) and for a situation in which the batch shop is split up into two smaller shops (S₂). Each of these two shops consists of two machines. It was found that a partitioning of J into J₁={1}, J₂={2} gives the best results on system performance. These results are mentioned in Table 10. In the settings (1-9) we consider job shops which consist of identical machines (1-3), machines with different capacities (4-6) and machines with machine dependent processing times (7-9). Note that for settings (7-9) average flow times are indicated instead of average waiting times.

<table>
<thead>
<tr>
<th>No</th>
<th>Factor</th>
<th>ρ</th>
<th>Average waiting (flow) time</th>
<th>Δ₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>default</td>
<td>0.3</td>
<td>2.67, 2.83</td>
<td>-6.0</td>
</tr>
<tr>
<td>2</td>
<td>default</td>
<td>0.6</td>
<td>4.21, 4.49</td>
<td>-6.7</td>
</tr>
<tr>
<td>3</td>
<td>default</td>
<td>0.9</td>
<td>9.02, 13.29</td>
<td>-47.3</td>
</tr>
<tr>
<td>4</td>
<td>capacities</td>
<td>0.3</td>
<td>2.79, 3.00</td>
<td>-7.5</td>
</tr>
<tr>
<td>5</td>
<td>capacities</td>
<td>0.6</td>
<td>5.02, 5.18</td>
<td>-3.2</td>
</tr>
<tr>
<td>6</td>
<td>capacities</td>
<td>0.9</td>
<td>10.63, 14.49</td>
<td>-36.3</td>
</tr>
<tr>
<td>7</td>
<td>service times</td>
<td>0.3</td>
<td>27.62, 28.25</td>
<td>-1.5</td>
</tr>
<tr>
<td>8</td>
<td>service times</td>
<td>0.6</td>
<td>29.63, 30.35</td>
<td>-2.4</td>
</tr>
<tr>
<td>9</td>
<td>service times</td>
<td>0.9</td>
<td>34.73, 39.76</td>
<td>-14.5</td>
</tr>
</tbody>
</table>

S₁ = a single job shop (N=2, M=4)
S₂ = two identical job shops (2 x N=1, M=2)
Δ₁ = 100\(^{(S_1 - S_2)/S_2}\)

Table 10 Average waiting (flow) time for S₁ : N = 2, M = 4 and S₂ : 2 x (N = 1, M = 2)

As can be concluded from Table 10 a split up of the job shop is not advantageous for either of the settings (see Δ₁). Here it is clear that DJAH is quite capable of handling the larger complexity. As might be expected from the results of queueing theory, performance differences are small for low traffic intensities, but high for high traffic intensities.

To illustrate the effect of a fixed allocation of machine groups to product type groups in case of product dominance, the two machine batch shop is split up into two smaller job shops of one machine each. In Table 11, results for DJAH are presented for the respective configurations (S₁, S₂). Note that given the restrictions set by machine characteristics, the best allocation of product types handled by each of the two job shops of S₂ may differ per setting. This is indicated in the Table by \((n₁, n₂)\). Hereby \(n₁\) represents the number of product types assigned to the first (settings 1-3), largest (settings 3-6) or fastest machine (settings 7-9). In the same way \(n₂\) represents the number of product types assigned to the other machine. For setting 9, no feasible allocation of machines to product types was found.

⁷ Such a resource allocation is covered by the first phase in our method (see Section 2).

⁸ Note that only identical product types are considered, i.e., processing times and maximum batch size do not depend on product type.
D.J. van der Zee, A. van Harten and P.C. Schuur

Average waiting time

<table>
<thead>
<tr>
<th>No</th>
<th>Factor</th>
<th>ρ</th>
<th>S₁,</th>
<th>S₂,</th>
<th>(nᵣ,nₓ)</th>
<th>Δᵣ</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>default</td>
<td>0.3</td>
<td>12.11</td>
<td>12.94</td>
<td>2,2</td>
<td>-6.9</td>
</tr>
<tr>
<td>2</td>
<td>default</td>
<td>0.6</td>
<td>18.16</td>
<td>19.54</td>
<td>2,2</td>
<td>-7.6</td>
</tr>
<tr>
<td>3</td>
<td>default</td>
<td>0.9</td>
<td>29.44</td>
<td>39.86</td>
<td>2,2</td>
<td>-35.4</td>
</tr>
<tr>
<td>4</td>
<td>capacities</td>
<td>0.3</td>
<td>12.40</td>
<td>13.65</td>
<td>2,2</td>
<td>-10.0</td>
</tr>
<tr>
<td>5</td>
<td>capacities</td>
<td>0.6</td>
<td>20.63</td>
<td>25.64</td>
<td>3,1</td>
<td>-24.3</td>
</tr>
<tr>
<td>6</td>
<td>capacities</td>
<td>0.9</td>
<td>35.03</td>
<td>67.17</td>
<td>3,1</td>
<td>-91.7</td>
</tr>
<tr>
<td>7</td>
<td>service times</td>
<td>0.3</td>
<td>37.28</td>
<td>37.91</td>
<td>3,1</td>
<td>-1.7</td>
</tr>
<tr>
<td>8</td>
<td>service times</td>
<td>0.6</td>
<td>43.39</td>
<td>45.06</td>
<td>3,1</td>
<td>-3.8</td>
</tr>
<tr>
<td>9</td>
<td>service times</td>
<td>0.9</td>
<td>54.96</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

S₁ = a single job shop (N=4, M=2)
S₂ = two job shops (N=nᵣ, M=1)
(nᵣ,nₓ) = the number of product types handled by either of the two job shops of S₂
Δᵣ = 100*(S₁ - S₂)/S₁.

Table 11 Average waiting time for S₁: N=4, M=2 and S₂: 2 x (N = 2, M = 1)

The outcome of the simulations is in accordance with the results found for the example situation discussed earlier in this subsection. Note the importance of allocating the machines to the right number of product types.

7. Concluding remarks and suggestions for further research

Our main conclusions are:

- **Effective look-ahead strategies for multi-machine, multi-product batching processes can be constructed in a systematic way.** In many industries such as the aircraft industry or the semi-conductor industry, multi-server batch processing machines are found. These systems are often composed of multiple alternative machine types. Extension of existing look-ahead strategies to this type of systems appeared to be relatively simple, using a newly developed method, see Section 2. This method concerns a construction method for look-ahead strategies. While on the one hand it guides the construction process, it facilitates a better understanding of the problem situation on the other hand.

- **There is a clear difference between control of machine dominant and product dominant shops.** By a series of simulation experiments the potential of three new heuristics has been tested for both systems consisting of identical machines and systems consisting of non-identical machines. One of the basic questions was: whether the inclusion of other machines in the decision would improve system performance. For situations in which the number of machines is greater than the number of product types, a relatively greedy heuristic shows the best performance. This heuristic concerns a straightforward extension of the Dynamic Job Assignment Heuristic (DJAH). The heuristic adopts a short cost horizon (up to the moment the next machine becomes available). Besides, it does not involve alternative machines in its decision. However, for cases in which the number of product types is dominant, heuristics which include other machines in the decision indicate a better system performance.

- **Our look-ahead strategies can handle complexity so well, that complexity reduction by machine grouping leads to performance loss.** The concept of machine grouping, i.e., a fixed allocation of machine groups to product groups, given the application of a look-ahead strategy has been tested in several settings. It has been found that machine grouping strongly reduces system performance, as it throws away the greater flexibility of the larger system.

Several interesting suggestions for future research on look-ahead strategies for batch shop control can be given, which relate to:

- **Different system characteristics**, e.g. systems in which multiple processing steps are foreseen (c.f. Robinson et al. (1995) who describe a batch-serial system and Glassey et al. (1993) who study re-entry flows in batch shop environments). Other extensions relating to practical situations are the limitation of buffer capacity, the possibility of machine breakdowns, the forming of families of product groups with different capacity requirements per unit of product and quality constraints which restrict the possibility to postpone loading of products (c.f. Hodes et al. 1992).
• Other cost structures/performance criteria. Other types of costs are sequence-dependent setup costs and penalty costs for late deliveries. Alternative performance criteria may be based on due date settings or possibilities to prioritize the processing of certain product types (for example because they are needed urgently elsewhere).

References