Proceedings of the 2008 Winter Simulation Conference
S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler eds.

TIME-LIMITED NEXT ARRIVAL HEURISTIC FOR BATCH PROCESSING AND SETUP REDUCTION IN A RE-ENTRANT ENVIRONMENT

Stephen Murray
John Geraghty
Paul Young

Enterprise Process Research Centre (EPRC)
School of Mechanical and Manufacturing Engineering
Dublin City University
Glasnevin, Dublin 9, IRELAND

Steve Sievwright

Intel Ireland (IFO)

Collinstown Industrial Park
Leixlip, Co. Kildare, IRELAND

ABSTRACT

This paper presents a new batch scheduling heuristic - the Time-Limited Next Arrival heuristic for batch processing and setup reduction (TLNA). This heuristic has been defined for a batch processing machine group in a re-entrant manufacturing environment where setups are sequence-dependent. When making the scheduling decision, TLNA takes into account future arrivals based on a user-defined wait time. A series of experiments is conducted on a discrete event simulation model to determine the impact of this wait time. A total cost function is used to combine two conflicting performance measures (total item queuing time and total machine running time) into one. All TLNA wait time scenarios are compared to the Next Arrival Control Heuristic for Multiple products and Multiple machines (NACHMM). The experiments presented show that there is a wait time that minimises the total operational cost. TLNA outperforms NACHMM with regard to all performance measures except total queuing time.

1 INTRODUCTION

Batch scheduling involves the allocation of items to machines that can process more than one item simultaneously in a batch. The key question is whether to start running the machine with a less-than-full batch or wait for the next compatible item to arrive before processing. There is, therefore, a trade-off between queuing time reduction and utilisation increase at the batch processing machines. The scheduling decision is made easier if knowledge of the future item arrivals is available and this may be extracted from most CIM systems.

Many different techniques have been used to solve the batch scheduling problem. Due to the complexity of these problems and the extensive computational time required to find an optimal solution; a very popular choice has been the use of heuristics that seek good, if not optimal, solutions in a relatively short space of time.

Look-ahead heuristics for batch scheduling can be divided into a number of categories: those for the single product – single machine case (Glassey and Weng 1991; Gupta and Sivakumar 2006), multiple product – single machine case (Fowler, Phillips, and Hogg 1992; Weng and Leachman 1993; Duenyas and Neale 1997) and multiple product – multiple machine case (van der Zee, van Harten, and Schuur 1997; Fowler, Hogg, and Phillips 2000; Cigolini et al. 2002). Some heuristics have also been developed which incorporate a downstream stage into the scheduling decision (Robinson, Fowler, and Bard 1995; van der Zee 2002; Solomon et al. 2002).

Most of this research looked at cycle time or queuing time as a performance measure; although machine utilisation, holding cost and due date-based measures were also considered. However, in all of these studies the cost of operating the machines was not included. Also, very little has been reported in the literature on batch scheduling heuristics with respect to techniques that simultaneously address conflicting performance measures (e.g. the trade-off between queuing time and utilisation in batch processing). This research attempts to address the trade-off between the conflicting desires to reduce queuing time and reduce the total machine running (setup + processing) time. In this way, the performance of each experimental scenario can be assessed in the knowledge that one measure is not having a detrimental effect on the other.

Another important aspect that appears to be neglected for the most part in the work discussed above is sequence-dependent setup times whereby the time to change a machine over to another item type depends on the characteristics of both the previous and current item processed. Where setup times have been incorporated, these are generally sequence-independent and included as part of the processing time. However, according to Allahverdi and

978-1-4244-2708-6/08/$25.00 ©2008 IEEE

Soroush (2008), treating setup times separately from processing times would allow operations to be performed simultaneously and hence improve resource utilisation. Uzsoy et al. (1991) agree by stating that ignoring setup considerations in scheduling decisions can lead to excessive amounts of time being spent in setup changes. Therefore, any successful scheduling system must take the sequence-dependent nature of setups into account.

This paper considers four main objectives: 1) to develop a heuristic for the batch scheduling of a specific, real-life, machine group problem, taking the sequence-dependent nature of the setups into account, 2) to incorporate a look-ahead function into this heuristic which makes use of the knowledge of future arrivals, 3) to combine two conflicting objectives (reduce both queuing time and machine running time) into one (cost-related) performance measure, and 4) to experiment with variations of this heuristic and compare it to an existing generic heuristic through a simulation model of the machine group.

The remainder of this paper is structured as follows: section 2 presents the machine group that is used as the test-bed for the simulation experiments. Section 3 details both the new heuristic and the NACHMM heuristic used as a baseline measure of performance. Section 4 discusses the simulation experiments and is followed by the results in section 5. Finally, section 6 outlines the research conclusions and directions for future work.

2 THE MACHINE GROUP

The machine group under study (Figure 1) is one of a number in the factory that is used as a test-bed. When an item is released into the line it is assigned a route that dictates its series of operations. Re-entrancy is exhibited which means that when items finish processing at a machine group, they move on to their subsequent operations before returning for their next operation at that group.

The machine group in question consists of identical machines in parallel which can all process the same set of more than 50 different operations. A buffer is situated in front of the machines and feeds them in the order dictated by the heuristics described in section 3. Each machine can be loaded with either one item or two items in a batch where the mean processing time for a single and double-loaded machine is the same. If a batch is loaded onto a machine, the two items within that batch must be of the same operation.

The operations may be characterised as one of three families (A, B or C), each requiring a different machine configuration. The machines are subject to sequence-dependent setup times. These are especially short when processing the same operation as was on the machine previously, become longer when changing over to a different operation of the same family, and longer still when the machine must be configured to process an operation of a different family. Clearly the first type of setup is most desirable with the last type being least desirable.

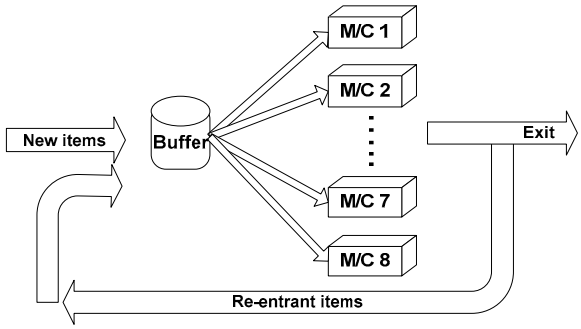

Figure 1: Schematic of the machine group

A cost per hour is associated with the items queuing in the buffer. This cost is only 10% that of running one of this group's machines per hour, derived from energy and resource requirements. A machine is said to be running if it is either in a setup or processing state. When the machines are idle they are automatically put on standby. This reduces their energy and resource consumption by 50%, reducing the cost per hour of a machine on standby to half that of a running machine. The machines can also be shut down completely. This reduces their energy and resource consumption to 0, meaning that the cost per hour of a shut down machine is $0. However, unlike machines in a shut down state, machines in a standby state remain calibrated and can be brought up to production immediately. A large cost is involved with bringing a shut down machine up to production as a number of skilled technicians and pieces of expensive equipment are required. This process can take several days to complete so the decision to shut down a machine must be made carefully.

This type of problem is relevant to any production system which is characterised by some if not all of the following: parallel processes, re-entrancy, batch processing and sequence-dependent setups. Traditionally such systems are used in semiconductor fabrication and batch pharmaceutical production.

3 THE SCHEDULING HEURISTICS

There are two existing heuristics that could be applied to this problem, the first being Fowler, Hogg, and Phillips' (2000) Next Arrival Control Heuristic (NACH) for Multiple products and Multiple machines (referred to as NACHMM hereinafter). The second one is the multiple product – multiple machine case of Cigolini et al.'s (2002) Wait No Longer Than Time (WNLTT) heuristic.

NACHMM was created to control the diffusion and oxidation processes in semiconductor manufacturing with the aim of reducing the average lot queuing time as part of an industry focus on overall cycle time reduction. It com-

pares processing a less than full batch with completing the batch before processing, on the basis of the queuing time for all affected lots. If more time can be saved with the latter case then the machine is idled until the next arrival.

NACHMM is more complex than TLNA in that it uses a series of detailed calculations to prioritise the lots and more information is taken into account during the selection process (number of idle machines, time that a machine will next become available, next arrival time of any product). Also, NACHMM is parameter-free whereas TLNA requires the user-defined wait time in order to function correctly. Perhaps most significantly, the individual setup times for each operation are included in the TLNA calculation but not in NACHMM, whereas the opposite is true for processing times.

WNLTT works in a similar way to NACHMM but rather than looking at specific lot arrivals, it uses a time window within which a batch must be loaded onto an idle machine. This time window is recalculated every time an event occurs (push or pull – same principles as in section 3.1) and decreases in length every time a lot arrives.

Like NACHMM, WNLTT is more complex than TLNA and has as its main objective cycle time reduction. TLNA uses a definite wait time when looking ahead to future lot arrivals while WNLTT uses a dynamic window.

NACHMM is chosen as the heuristic for comparison because of the fact that, when making the scheduling decisions, its use of only the next arrival of each product closely resembles TLNA and also because of its ease of application to this problem. In applying NACHMM here, products are replaced by operations and preference for operations within the same family is added. This allows the use of the NACHMM heuristic as the baseline in the simulation experiments.

The following section describes the new Time-Limited Next Arrival heuristic for batch processing and setup reduction (TLNA). In contrast to the heuristics outlined above, TLNA is relatively straightforward which should lead to easy implementation in the factory. The combined measure of performance used to assess TLNA also differs from the other heuristics which primarily look at cycle time-related performance measures.

3.1 Time-Limited Next Arrival Heuristic for Batch Processing and Setup Reduction (TLNA)

This heuristic takes into account the quantity and operation of items currently queuing in the buffer and the arrival time of the next item of each operation. There are two types of decision moments: push and pull. A push decision moment occurs when an item enters the buffer and a pull decision moment occurs when a machine finishes processing its current item. This is illustrated in Figure 2 and the scheduling logic involved in the two decision moments is detailed in sections 3.1.1 and 3.1.2.

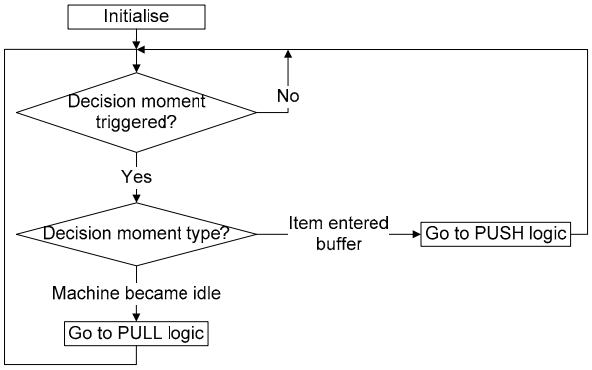

Figure 2: TLNA decision moments

The heuristic aims to both reduce setup times and increase the number of two-item loads. It achieves a reduction in setup times by giving a higher priority to shorter setups. It achieves an increase in the number of two-item loads by giving a higher priority to those operations that have more than one item queuing in the buffer. Setup times are short in comparison to processing times, so making a two-item load has a higher priority than running an operation which results in a short setup time.

This is a very specific and complex problem so an existing 'off-the-shelf' look-ahead batching heuristic may not yield the best results. To determine whether or not this is the case, TLNA is compared to the heuristic deemed to be most appropriate for comparison, NACHMM.

Nomenclature for both Push & Pull Logic Heuristics

B :	the batch size, where $0 < B <= 2$
Op :	the new item's operation
F :	Op's family
Q :	the number of items of Op now queuing in the buffer
M :	the set of idle and eligible machines
$PrevOp$:	the last operation processed on the idle machine
$PrevF$:	$PrevOp$'s family
T :	the set of eligible and unassigned items queuing in the buffer

3.1.1 Push Logic

A flow diagram of TLNA's push logic is illustrated in Figure 3 (see next page). The general order of priority dictated by the logic can be seen in the shaded rows of Table 1. When a machine is idle, a two-item load (either ready to load now or the second item of the load is within the user-defined wait time) of the operation that was last processed on this machine has the highest priority. This is followed by a two-item load of the same family and so on. A one-item load of an operation belonging to a different family to the one that was last processed has the lowest priority. This demonstrates the point made in section 3.1

that two-item loading has priority over setup reduction in the heuristic.

Table 1: Example of the TLNA push logic in action

Machine	**M/C 3**	**M/C 4**
Idle / Busy	Idle	Busy
Last / Current Operation	51	35
Last / Current Family	C	B
Same Operation (2-item load or wait)	0	0
Same Family (2-item load or wait)	0	0
Different Family (2-item load or wait)	0	0
Same Operation (1-item load)	0	0
Same Family (1-item load)	0	**19**
Different Family (1-item load)	**19**	0
Next Operation	-	**19**

There are a number of instances in the logic where *M* can be chosen from a number of eligible machines. Ties are broken by assignment to the tool with the lowest identification number.

If *Op* is 'assigned' to another (currently busy) machine, this means that that machine's desire to process the operation is greater than that of one of the currently idle machines. Table 1 illustrates this point by using an example focusing on two of the machines. It shows that the last operation to run on idle machine 3 was 51 (family C) while the operation currently running on busy machine 4 is 35 (family B). In this example, an item of operation 19 (family B) arrives into the currently empty buffer. The only idle machine in the group is machine 3. Following the push logic, machine 3 desires to run this newly arrived item. The loading type for this machine has the lowest priority in the logic as it is a one-item load of a different

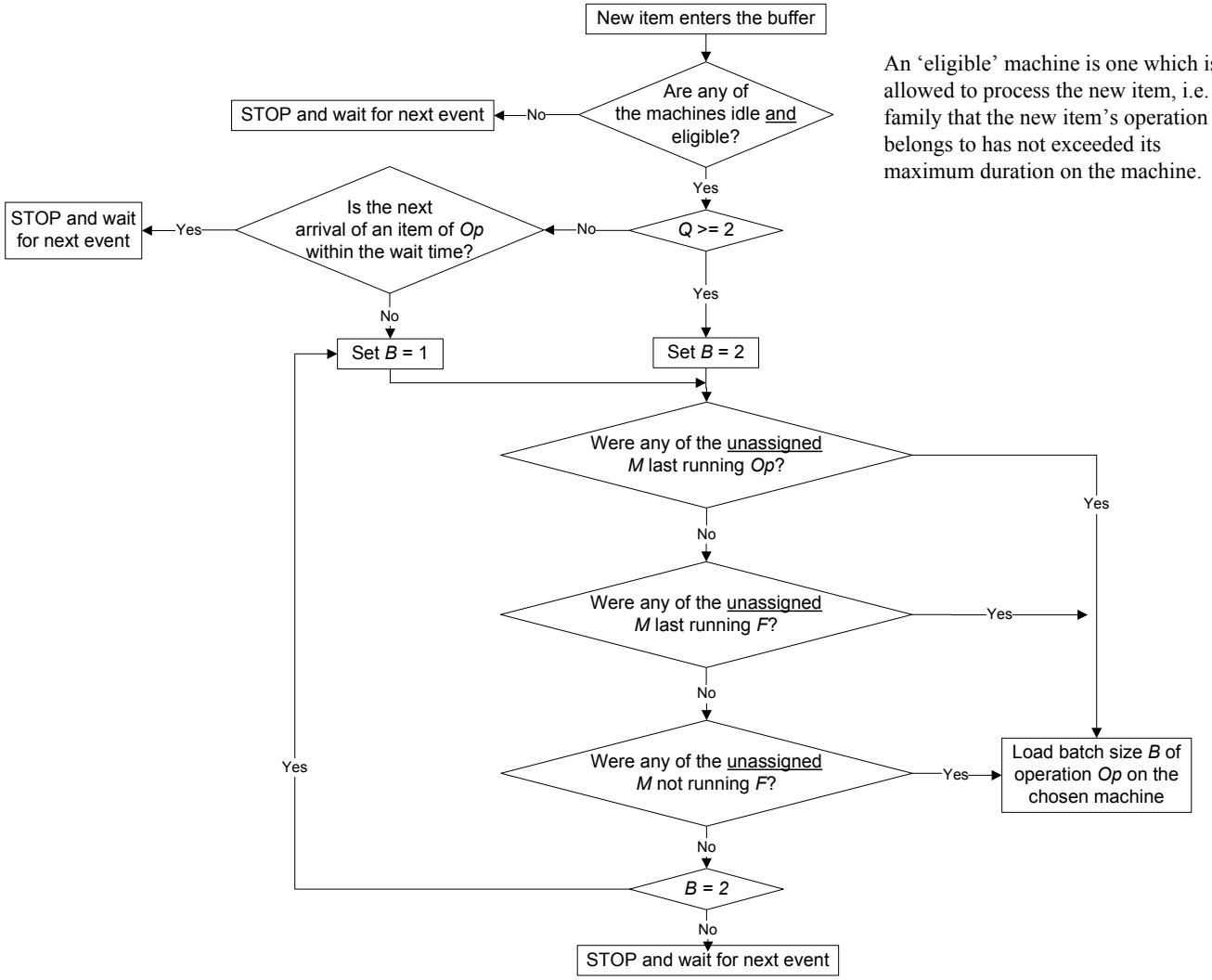

Figure 3: The TLNA push logic

family. Machine 4 also desires to run the new item, but its loading type (one-item load of the same family) has a higher priority than machine 3's.

Machine 3, therefore, remains idle until the next push decision moment whereas machine 4 will be loaded with this item when it idles. This demonstrates that when machines are busy they influence the scheduling of the idle machines so as to increase the number of two-item loads they process and, as in this case, reduce their setup times.

3.1.2 Pull Logic

A flow diagram of TLNA's pull logic is illustrated in Figure 4. It's nomenclature can be found in section 3.1.

The pull logic works in the same general way as shown in Table 1. However, using that example, instead of an item of operation 19 arriving in the buffer, the decision moment occurs when machine 3 becomes idle and there is only that one item queuing. Machine 3 is kept idle until the next push decision moment while machine 4 is loaded with this item when it becomes idle.

There are some rare instances in the pull logic where the operation to be loaded can be chosen from a number of eligible operations. In this case, the lowest operation number is selected. For example, if both operation 3 and operation 32 are eligible for loading, the item(s) of operation 3 are loaded.

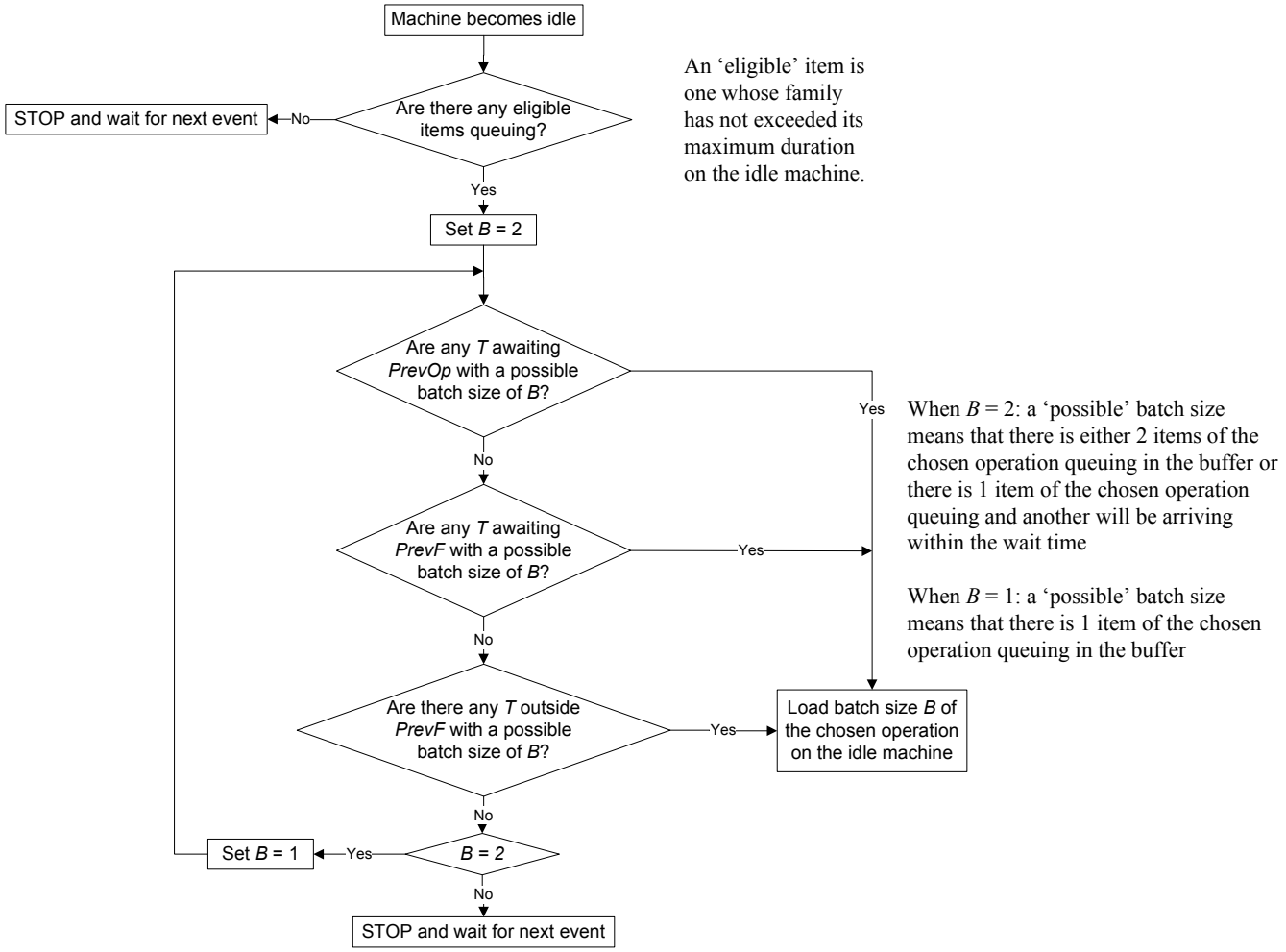

Figure 4: The TLNA pull logic

4 THE SIMULATION EXPERIMENTS

The parameter investigated through the simulation experiments is the TLNA wait time value introduced in sections 3.1.1 and 3.1.2. In this way it can be determined how long to wait, or 'look-ahead,' for an item of each operation when making the scheduling decision. As a result of looking ahead, certain machines can be held idle until the arrival of the item that means a two instead of one-item loading, a shorter setup time, or both.

The performance measure of interest is the total cost, made up of the queuing time and machine running time cost elements detailed in section 2. Therefore, both the queuing time and running time are investigated for each scenario so that the total cost can be calculated. This total cost measure enables two conflicting measures of performance to be combined into a multi-criteria solution. Running one-item loads results in shorter overall queuing time but longer machine running time, as items are not held in queue but rather are pulled into idle machines immediately. On the other hand, holding items in queue will of course increase the queuing time but enable machines to idle for longer. Assigning a cost to both the queuing time per hour and machine running time per hour means that the two measures can be assessed simultaneously.

For the purposes of this study a period representing approximately half the average processing time for an operation was selected as the basis for the different look-ahead time limits. Using one period effectively limits the heuristic to consideration of items no more than one upstream operation from the machine group.

Some items within that period may be queuing or processing at the previous operation, or may be in transit to the buffer for this machine group but crucially, they are no further than one operation away. For experimentation, increments of 2 periods are used to effectively expand the look-ahead time by an operation for each increment, to a maximum of 9 periods. The NACHMM heuristic with its own look-ahead function is applied to the problem as the baseline scenario through which all TLNA scenarios can be compared. Seven heuristic scenarios are applied:

1. NACHMM (the baseline scenario)
2. TLNA: No wait (no look-ahead)
3. TLNA: wait 1 period (1-operation look-ahead)
4. TLNA: wait 3 periods (2-operation look-ahead)
5. TLNA: wait 5 periods (3-operation look-ahead)
6. TLNA: wait 7 periods (4-operation look-ahead)
7. TLNA: wait 9 periods (5-operation look-ahead)

The model is populated with initial conditions at the simulation start time. Therefore, no pre-steady state bias has to be removed before recording the output results. 30 runs of each scenario have been conducted to yield statistically significant results. The method of common random numbers (Law 2007) is used to compare scenarios.

5 RESULTS AND DISCUSSION

This section is laid out as follows: first, the queuing time performance is considered for each scenario, followed by the machine running time. From that, the total cost values are calculated and compared and the scenario which results in the lowest total cost is identified.

5.1 Total Queuing Time

Figure 5 shows the percentage total queuing time endured over the duration of the week for each TLNA scenario w.r.t. the NACHMM heuristic.

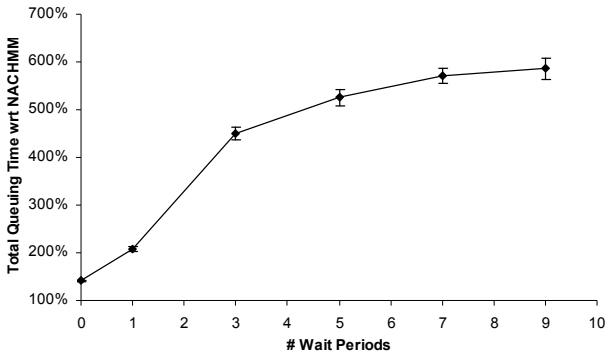

Figure 5: Total queuing time for each scenario w.r.t. NACHMM

It is evident from Figure 5 that the item queuing time increases exponentially with respect to the user-defined wait time. When there is no look-ahead, the items are loaded onto the machines regardless of whether they are in a batch not. As the wait time increases, there is a larger horizon within which to create batches so there is more chance of an item being held in the buffer until a match arrives. The slope becomes less steep after the 3 wait period mark so after this point the effect of increases in the wait time value is lessened. NACHMM gives the lowest queuing time result which indicates that it succeeds in its aim to reduce this performance measure.

5.2 Total Machine Running Time

Figure 6 shows the percentage total machine running time (setup + processing time) over all machines for each TLNA scenario w.r.t. the NACHMM heuristic. It is evident from Figure 6 that as the wait time increases, the machine running time decreases. This means that the items are being held in the queue waiting for another item with which to make a batch while machines are left idle. When there is no look-ahead, the idle machines pull in whatever is in the buffer regardless of whether or not a batch will be made. The slope lessens for increases in wait time and

NACHMM yields the highest total running time which is consistent with the queuing time results.

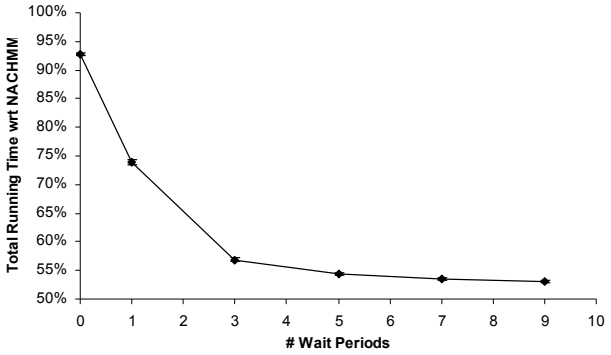

Figure 6: Total machine running time for each scenario w.r.t. NACHMM

5.3 Total Cost

From section 2, the cost per hour for a single item queuing in the buffer is 10% that of running one of the machines per hour. The cost per hour of a machine on standby is 50% that of the running cost. Based on those figures the total cost per scenario is calculated from:

$$C = C_q \times \sum_{i=1}^{n} QT_i + C_r \times \sum_{j=1}^{m} RT_j + C_s \times \sum_{j=1}^{m} ST_j \qquad (1)$$

where:

- C = total cost
- C_q = cost/hr of a queuing item
- C_r = cost/hr of a running machine
- C_s = cost/hr of a machine on standby
- QT_i = total queuing hours for item i
- RT_j = total running hours (setup + processing time) for machine j
- ST_j = total standby hours (time spent not in setup or processing an item) for machine j
- n = the total number of items to have entered the machine group
- m = the total number of machines

Figure 7 shows the % total cost (using (1)) created by the machine group over the week for each TLNA scenario w.r.t. the NACHMM heuristic. It is clear that in this experimental set-up, scenario 2: TLNA with a wait time of 1 period (looks one operation ahead), yields the lowest total cost out of the scenarios considered. There is a sharp decrease of approximately 6% when increasing the wait time to 1 period. After a small increase in cost for the 3 periods wait time, the cost begins to increase exponentially for every increase in wait time. However, the performance when the wait time is 9 periods is still better than not looking ahead. NACHMM performs the most poorly out of all of the scenarios, including the one that does not look upstream.

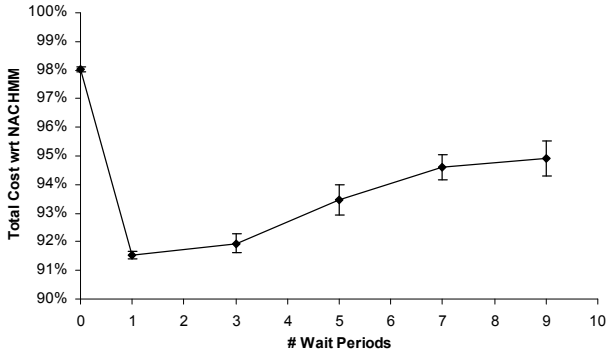

Figure 7: Total cost for each scenario w.r.t. NACHMM

The fact that the cost per hour of running a machine is ten times greater than the cost per hour of queuing an item has a heavy influence on the total cost results. If these costs were equal the emphasis would be placed on reducing the queuing time. This is because the amount of queuing time hours for the high wait time scenarios is much greater than the amount of machine running hours in the low wait time scenarios (see Figures 5 and 6).

The variation in the total number of items processed across all the models is statistically insignificant. This indicates that the performance of the factory would not be compromised with respect to throughput when TLNA is implemented at this machine group.

6 CONCLUSIONS AND FUTURE RESEARCH

This paper considers a specific machine group batch scheduling problem with characteristics applicable to many different industries. Batching and sequence-dependent setups make this problem extremely complex.

The Time-Limited Next Arrival heuristic for batch processing and setup reduction has been proposed which prioritises and schedules queuing items based on their operation, family and the time until the next arrival of the same operation. The heuristic aims to first increase the number of two-item batches created and secondly to reduce setup times. TLNA has a look-ahead function included in the logic whereby items are held in the buffer if another item of the same operation is set to arrive within a user-defined wait time. This part of the logic increases the potential number of two-item batches which may be created.

A comparison of the TLNA heuristic with NACHMM based on their performance when applied to a discrete event simulation model, which adds uncertainty to the complexity outlined above, has been presented. Of the experimental results, the best results have been obtained for a wait time of 1 period.

Every one of the look-ahead scenarios outperforms the no wait time case in terms of total cost. This is congruent with previous research into batch scheduling which specify the importance of using the knowledge of future arrivals when making scheduling decisions. Every TLNA scenario (including the non-look-ahead case) outperforms NACHMM with regard to total cost which shows that using knowledge about the problem can lead to better results than using a generic heuristic which may only be suited to basic problems. Also, because NACHMM focuses on cycle time as a single performance measure, it outperforms TLNA in that regard but suffers greatly when it comes to reducing the machine running time. Therefore, the total cost of running NACHMM in this situation is considerably greater than TLNA. This validates the use of a combined performance measure when assessing different scenarios. Using NACHMM for scheduling in this machine group would be extremely myopic, i.e. queuing time would be greatly reduced but to the detriment of the group as a whole.

The principle implication of the results is that a trade-off does exist, in batch processing environments with sequence-dependent setups, between the cost of queuing time and the cost of utilisation. On this basis, the recommendation for the specific problem studied in this article is that the scheduler should take into account items one operation, but no more, upstream when making loading decisions while the cost characteristics outlined in the paper hold true. As the 1 period wait time is less than the average processing time of the upstream operation, this recommendation is valid. If, however, this average processing time was reduced without adjusting the wait period, the recommendation laid out here may have to be revisited in that a period may represent more than a one operation look-ahead.

The focus of future research will extend the problem to include batch-dependent processing times, individual item priorities, and forecast errors with regard to the arrival times of items, to test the robustness of the heuristic in the face of increasing uncertainty. It is also intended to experiment with the heuristic under other production regimes (e.g. traffic intensity, product mix and idle cost to running cost ratio) which will influence the order in which items are processed. Extension of the analysis to include other performance criteria such as service level, item tardiness and throughput will incorporate the real goal of manufacturing; the delivery of finished goods to customers.

ACKNOWLEDGEMENTS

The authors wish to thank the industrial partner for their dedicated support during the project. Special appreciation is expressed to Dr. Arijit Bhattacharya for his extensive suggestions and help in preparing this manuscript. This research is funded by the Embark Initiative from the Irish Research Council for Science, Engineering and Technology (IRCSET) and the National Development Plan (Grant Reference Number: RS/2005/93).

REFERENCES

Allahverdi, A,. and H. M. Soroush. 2008. The significance of reducing setup times/setup costs. *European Journal of Operational Research* 187:978-984.

Cigolini, R., M. Perona, A. Portioli, and T. Zambelli. 2002. A new dynamic look-ahead scheduling procedure for batching machines. *Journal of Scheduling* 5:185-204.

Duenyas, I., and J. J. Neale. 1997. Stochastic scheduling of a batch processing machine with incompatible job families. *Annals of Operations Research* 70:191-220.

Fowler, J. W., G. L. Hogg, and D. T. Phillips. 2000. Control of multiproduct bulk server diffusion/oxidation processes. Part 2: Multiple servers. *IIE Transactions (Institute of Industrial Engineers)* 32:167-176.

Fowler, J. W., D. T. Phillips, and G. L. Hogg. 1992. Real-time control of multiproduct bulk-service semiconductor manufacturing processes. *IEEE Transactions on Semiconductor Manufacturing* 5:158-163.

Glassey, C. R., and W. W. Weng. 1991. Dynamic batching heuristic for simultaneous processing. *IEEE Transactions on Semiconductor Manufacturing* 4:77-82.

Gupta, A. K., and A. I. Sivakumar. 2006. Optimization of due-date objectives in scheduling semiconductor batch manufacturing. *International Journal of Machine Tools and Manufacture* 46:1671-1679.

Law, A. M. 2007. *Simulation modeling and analysis*. 4th ed. McGraw-Hill, Inc.

Robinson, J. K., J. W. Fowler, and J. F. Bard. 1995. Use of upstream and downstream information in scheduling semiconductor batch operations. *International Journal of Production Research* 33:1849-1869.

Solomon, L., J. W. Fowler, M. Pfund, and P. H. Jensen. 2002. The inclusion of future arrivals and downstream setups into wafer fabrication batch processing decisions. *Journal of Electronics Manufacturing* 11:149-159.

Uzsoy, R., L. A. Martin-Vega, C. Lee, and P. A. Leonard. 1991. Production scheduling algorithms for a semiconductor test facility. *IEEE Transactions on Semiconductor Manufacturing* 4:270-280.

Van Der Zee, D. J. 2002. Adaptive scheduling of batch servers in flow shops. *International Journal of Production Research* 40:2811-2833.

Van Der Zee, D. J., A. Van Harten, and P. C. Schuur. 1997. Dynamic job assignment heuristics for multi-server batch operations - a cost based approach. *Inter-*

national Journal of Production Research 35:3063-3093.

Weng, W. W., and R. C. Leachman. 1993. An improved methodology for real-time production decisions at batch-process work stations. *IEEE Transactions on Semiconductor Manufacturing* 6:219-225.

AUTHOR BIOGRAPHIES

STEPHEN MURRAY is a Ph.D. student in the School of Mechanical and Manufacturing Engineering at Dublin City University in Ireland. He is collaborating with Intel Ireland for his Ph.D. project. His research interests include discrete event simulation and optimization of semiconductor manufacturing systems, specifically in the area of scheduling. His e-mail address is <stephen.murray5@mail.dcu.ie>.

STEVE SIEVWRIGHT is 'Manufacturing Sciences Project Manager' at Intel's 200mm wafer fab in Leixlip Ireland. He has a first class BSc degree in 'Electronics and Microcomputer Systems' from the University of Dundee, and a masters degree in 'Manufacturing Management' from the Open University (UK). His email address is <steve.d.sievwright@intel.com>.

JOHN GERAGHTY joined Dublin City University in 2003 after completing his PhD thesis at the University of Limerick on the performance of several production control strategies, such as Kanban, CONWIP, Hybrid Kanban-CONWIP etc, in the presence of uncertainty in customer demand. Current research interests include Operations Excellence and Supply-Chain Management. His e-mail address is <John.Geraghty@dcu.ie>.

PAUL YOUNG joined Dublin City University in 1998. Following his PhD (Dublin University, 1991) on monitoring the turning process, he worked on vehicle NVH research. Since 1995 he has been active in manufacturing research looking at process monitoring and simulation of the manufacturing system. He is a founder member of the Enterprise Process Research Centre in DCU. His email address is <Paul.Young@dcu.ie>.