Batch scheduling in a two machine flow shop with limited buffer and sequence independent setup times

Marco Pranzo

Abstract— **The problem of makespan minimization in a flow shop with two machines when the input buffer of the second machine can only host a limited number c of parts is known to be NP-hard for any $c > 0$ and $c \leq n$, where n is the number of jobs. In this paper we analyze this problem in the context of batch scheduling, i.e, when identical parts must be processed consecutively. In particular we study the case in which each batch requires sequence independent initial and/or final setup times. We then show that, if the size of the i-th batch is larger than a value b_i^*, then the makespan minimization problem can be formulated as a special case of TSP and solved in polynomial time. The cost structure of this TSP can be reduced to the one defined for the two machine no-wait flow shop. Hence, we give a closed form expression for b_i^*. Then, we prove that when the same algorithm is applied to batch sizes smaller than b_i^*, the error goes to zero as the batch sizes approach the values b_i^*.**

Keywords—

I. Introduction

SEVERAL production environments are organized according to the philosophy of the flow lines: machines are arranged along a line, raw parts enter the line from one endpoint, and finished products leave the line from the other endpoint. All the parts visit the machines in the same order. Flow lines have inherent advantages, such as low cost, ease of supervision etc. (for a thorough description of flow lines see for instance the book by Askin and Standridge [2]).

A considerable amount of literature has been written concerning optimization models in flow lines. The two major problems arising in the management of flow lines are the assignment of the operations of the jobs to the machines of the line, and the problem of sequencing the jobs on the machines. The former problem is addressed by the ASSEMBLY LINE BALANCING model. It consists of assigning a given set of operations to the minimum number of machines in the line, while respecting a constraint on the cycle time. The latter problem is referred to as FLOW SHOP SCHEDULING, and consists of finding the sequence of jobs on the machines of the line such that the overall completion time is minimized.

Although a very large number of papers have been written on flow shops, only a few authors have dealt with one of the major issues of real-life systems, namely the limited capacity of the intermediate storage facilities. One of the reasons is probably the fact that even when the line consists of two machines only, if the capacity of the intermediate buffer is limited, the scheduling problem is already NP-hard. This was shown by Papadimitriou and Kanellakis [7].

On the other hand, while most scheduling algorithms are conceived to deal with the case in which the jobs are all different from each other, in practice parts are often produced in batches, each batch consisting of identical parts to be produced consecutively. Moreover batches need often an initial setup time or a final setup time in order to be processed by the machines. This happens, for example, in the steel making process.

The steel production process is well known to have extremely strict requirements of material continuity and flow time (see, for example Dorn and Shams [3], Lixin Tang et al.[6]). Figure 1 shows the layout of the production line for a stainless steel line located in central Italy.

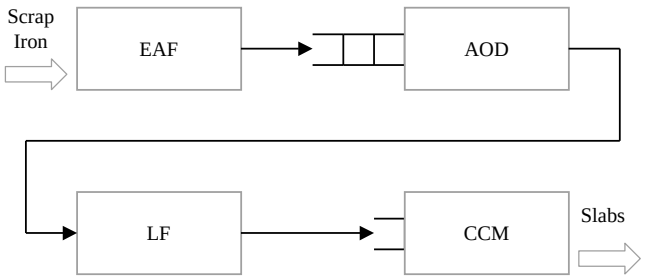

Fig. 1. Layout of the stainless steel production line

In particular, the processing of stainless steel consists of a sequence of high temperature operations starting with the loading of scrap iron in an Electric Arc Furnace (EAF). The time to melt the scrap iron is substantially independent of the batch type. The liquid steel is poured into ladles that a crane transports to a subsequent machine, called Argon Oxygen Decarburization unit (AOD), where nickel, chromium and other elements are added to the steel in order to meet the chemical quality requirements. After the AOD the ladles are transported to a Ladle Furnace (LF) which can host at most two ladles. Even if some operations are executed in the LF, in practice it acts as a buffer to maintain the ladles at the proper temperature before the last operation, to be executed in the Continuous Caster Machine (CCM). Here the liquid steel is casted and cooled to form slabs. The CCM needs to be tooled with a particular tool, called flying tundish, which must be changed when switching from a lot to another.

A first appoximated model of a stainless steel production line can be a batch scheduling problem in a two machine (AOD and CCM) flow shop, with an inital setup times given by the EAF operations on the first machine and a final setup times on the second machine, and with a limi-

Dipartimento di Informatica e Automazione, Università "Roma Tre" Via della Vasca Navale 79 - 00146 Roma, Italy E-Mail: mpranzo@dia.uniroma3.it .

tated intermediate buffer (LF).

We have studied a general model with initial and final setup times for each batch and each machine. These setup times are independent of the sequence but different for each batch. If the setup times are zero for each batch and each machine then the model collaps into a batch scheduling in a two machine flow shop with limited buffer problem [1].

The paper is organized as follows. In Section II we define the batch scheduling problem. In Section III we show that if the size of each batch B_i is greater than a certain quantity b_i^* (depending on the processing times of batch B_i and buffer capacity c), then the problem can be formulated as a TSP problem, and can be solved in polynomial time by defining an auxiliary instance of NO-WAIT FLOW SHOP. In Section IV we elaborate on the values of the b_i^*. In Section V we show that, even if $b_i < b_i^*$ for some i, the same TSP-based algorithm provides an ε-approximate solution, where ε approaches zero as the b_i's approach the values b_i^*. Throughout the paper, in all illustrations, the two bars correspond to machines M_1 and M_2, whereas the dashed lines indicate occupancy of buffer spaces (denoted by Q_1 and Q_2). All the illustrations refer to the case in which $c = 2$.

II. Problem definition

Consider a flow shop with two machines M_1 and M_2 and an intermediate storage buffer of capacity c, i.e., the buffer can host at most c parts. The system must process a set N of n batches, $B_1, B_2, \ldots, B_n$. Each batch B_i has size b_i, i.e., consists of b_i identical parts. All the parts from the same batch must be processed consecutively, i.e., batches cannot be split. Each batch B_i requires an initial (final) setup time U_{i1} (R_{i1}) and U_{i2} (R_{i2}), on the machine M_1 and M_2 respectively, independent of the sequence σ. The setup times are processed without the presence of one part on the machine. Each part of batch B_i requires processing times p_{i1} and p_{i2} on machines M_1 and M_2 respectively. For the sake of simplicity, let us first neglect the case in which $p_{i1} = p_{i2}$ for some B_i. This case will be discussed at the end of Section IV.

Let us partition N into two classes N_1 and N_2, where $N_1 \equiv \{B_i | p_{i1} < p_{i2}\}$ and $N_2 \equiv \{B_i | p_{i1} > p_{i2}\}$. If machine M_1 completes processing on a part and the buffer is full, then the part keeps M_1 blocked until there is room in the buffer. In this paper we suppose that $b_i \geq c + 1$ for all $i = 1, \ldots, n$.

As we already noticed, this condition is usually satisfied in real-life applications (consider that very often the buffer capacity c is itself very small). Notice that this condition implies that at any time in the system at most two batches can be under process. The problem we deal with in this paper can be formulated as follows:

Problem II.1 (Batch Sequencing Problem (BSP)) Given a two-machine flow shop with a buffer of capacity c, and a set of n batches $B_1, B_2, \ldots, B_n$, (batch B_i of size $b_i \geq c+1$, for $i = 1 \ldots n$, with setup times U_i and R_i, for $i = 1 \ldots n$, for each machine), find a sequencing σ of the batches so that the makespan is minimized.

Given a sequencing σ, we will denote as $\sigma(j)$ the j-th batch in the sequence. Problem II.1, which has been proved to be NP-hard by Agnetis et Al. [1] for any $c > 0$ and $c < k + 1$ even if all the setup time are zero, where k is the number of jobs. If $c = 0$, all the setup times can be viewed as part of the processing times, and therefore the problem can be reduced to the well-known NO-WAIT FLOW SHOP, which was solved in the two-machine case by Gilmore and Gomory [4], who provided polynomial time algorithm. If $c \geq k + 1$ and the setup times are zero, the buffer can be regarded as unlimited and hence the problem can be solved by the well-known Johnson's rule [5].

III. Makespan evaluation

In this section we show that, for a given solution σ to BSP, the value of the makespan can be expressed as the sum of two terms. The first term is the sum of quantities which do not depend on the sequencing of the batches, but only on the processing times and the batch sizes. The second term is the sum of quantities which do depend on the batch sequencing, and in fact this is what we want to minimize.

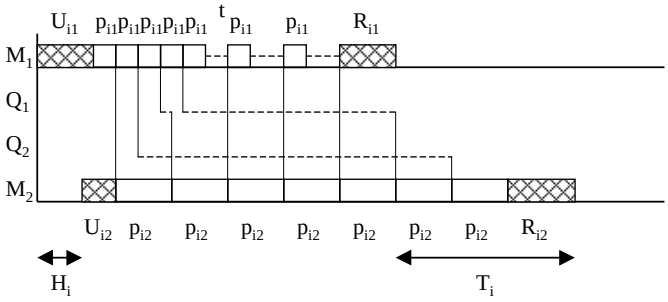

Fig. 2. $B_i \in N_1$ and steady state

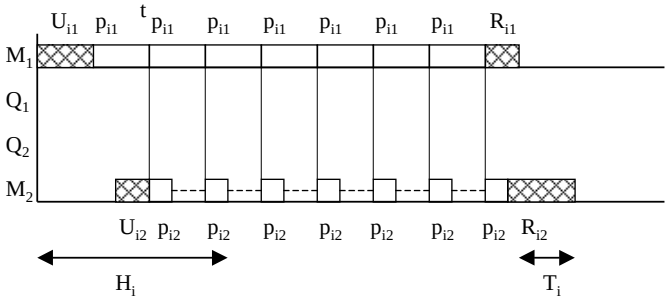

Fig. 3. $B_i \in N_2$ and steady state

Throughout the paper we use the concept of steady state, referred to the evolution of the schedule of a single batch when the size of B_i is large enough. As illustrated in figures 2 and 3, the steady state is reached at a time t such that, from then on, the two machines start processing a new part

at the same time, and this occurs every $\max\{p_{i1}, p_{i2}\}$ time units. In other words, after t the schedule of the batch repeats identically over time.

Given a solution σ, let us denote by $T(\sigma)$ the corresponding value of the makespan. Given a batch B_i, let us focus on the time interval from the start of the first part on M_1 and the release of the last part by M_1. The length of this time interval will be referred to as span of B_i, since it is the total amount of time that M_1 is busy with batch B_i (either operating on some part or blocked). In general, this quantity depends on the sequencing σ, and hence we denote the span of B_i as $S_i(\sigma)$. However, the value $S_i(\sigma)$ attains when B_i starts at time 0, (i.e., when the system is empty) obviously does not depend on σ, and therefore we denote it as S_i^0. We next compute this value, supposing that B_i reaches the steady state.

If $B_i \in N_1$, since $p_{i1} < p_{i2}$, the steady state is reached when the buffer is filled to its capacity. In this case, machine M_2 is never in a waiting state and therefore S_i^0 is equal to $S_i^0 = p_{i1} + (b_i - (c+1))p_{i2} + U_{i1} + R_{i1}$ (see figure 2). Note that on the first machine M_1 there is an idle time between the last part of the batch and the final setup time R_{i1}. In fact the first machine has to wait until the last part of the batch has moved into the first buffer Q_1. Only after that, the final setup R_{i1} can be processed on the first machine.

If $B_i \in N_2$, the buffer is always empty, the steady state is reached when the buffer is empty and $S_i^0 = b_i p_{i1} + U_{i1} + R_{i1}$ (see figure 3).

Let us introduce the concepts of head H_i and tail T_i of a batch B_i. We will define the head H_i of a batch as the maximum delay that does not produce an increase on the length of the makespan S_i, i.e., the delay is fully absorbed by the buffer. We will define the tail T_i of the batch B_i as the time interval between the end of the lavoration on the first machine M_1 and the end of lavoration on the second machine M_2. Note that the values of the head H_i and the tail T_i only depend on the batch B_i.

If $B_i \in N_1$ the head and the tail are given by

$$H_i = p_{i1} + U_{i1} - U_{i2} \tag{1}$$

$$T_i = p_{i2}(c+1) + R_{i2} - R_{i1} \tag{2}$$

If $B_i \in N_2$ the head and the tail are respectively

$$H_i = p_{i1}(c+1) + U_{i1} - U_{i2} \tag{3}$$

$$T_i = p_{i2} + R_{i2} - R_{i1} \tag{4}$$

In what follows, $\delta(x) = 1$ if $x > 0$ and $\delta(x) = 0$ if $x \leq 0$. The value of head and tail can be written as

$$H_i = p_{i1}(1 + c\delta(p_{i1} - p_{i2})) + U_{i1} - U_{i2} \tag{5}$$

$$T_i = p_{i2}(1 + c\delta(p_{i2} - p_{i1})) + R_{i2} - R_{i1} \tag{6}$$

Let us now suppose that B_i does not start at time 0, but rather B_i is scheduled immediately after some other batch B_h. Since M_2 and/or the buffer may not be empty at the time B_i starts on M_1, one has $S_i(\sigma) \geq S_i^0$. Therefore, $S_i(\sigma)$ can be expressed as $S_i^0 + Q(\sigma, i)$ with $Q(\sigma, i) \geq 0$. We call the two terms of the previous sum fixed cost and coupling cost respectively. We next show that if the preceding batch B_h and batch B_i have both reached the steady state, then the coupling cost only depends on the processing times of B_i and B_h, but it does not depend on the previous batches, i.e., $Q(\sigma, i) = Q(h, i)$.

Theorem III.1: Let B_h immediately precede B_i in a feasible solution σ. If both B_h and B_i reach the steady state, then the span $S_i(\sigma)$ is given by

$$S_i(\sigma) = S_i^0 + \max\{0,\ T_h - H_i\} \tag{7}$$

Proof: We divide the proof in two steps. The first step is divided in four subcases. For each subcase we want to show that the following expression

$$\begin{aligned} S_i(\sigma) = {S_i}^0 + \max\{0,\ & p_{h2}(1+c\delta(p_{h2}-p_{h1})) \\ & -p_{i1}(1+c\delta(p_{i1}-p_{i2}))-U_{i1}+U_{i2}+R_{h2}-R_{h1}\} \end{aligned} \tag{8}$$

holds.

1. $B_h \in N_1; B_i \in N_1$ (Fig. 4)
In this case the batch B_i is delayed by $\max\{0, p_{h2}(c+1) + R_{h2} - R_{h1} - p_{i1} - U_{i1} + U_{i2}\}$. This expression is in accordance with (8).

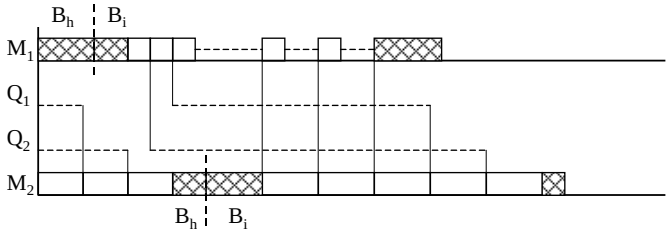

Fig. 4. $B_h \in N_1$ and $B_i \in N_1$

2. $B_h \in N_2; B_i \in N_1$ (Fig. 5)
In this case the batch B_i is delayed by $\max\{0, p_{h2} + R_{h2} - R_{h1} - p_{i1} - U_{i1} + U_{i2}\}$, and (8) holds.

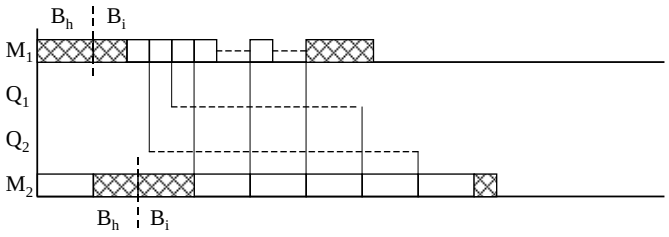

Fig. 5. $B_h \in N_2$ and $B_i \in N_1$

3. $B_h \in N_1; B_i \in N_2$ (Fig. 6)
In this case the batch B_i is delayed by $\max\{0, p_{h2}(c+1) + R_{h2} - R_{h1} - p_{i1}(c+1) - U_{i1} + U_{i2}\}$, and (8) holds.
4. $B_h \in N_2; B_i \in N_2$ (Fig. 7)
In this case the batch B_i is delayed by $\max\{0, p_{h2} + R_{h2} -$

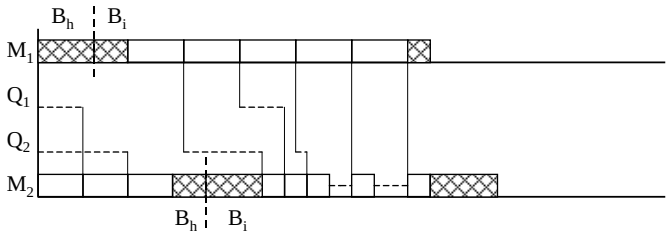

Fig. 6. $B_h \in N_1$ and $B_i \in N_2$

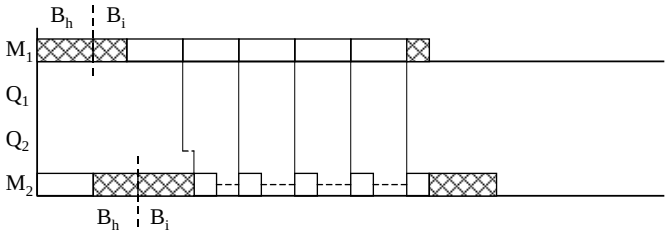

Fig. 7. $B_h \in N_2$ and $B_i \in N_2$

$R_{h1} - p_{i1}(c+1) - U_{i1} + U_{i2} = \max\{0, T_h - H_i\}\}$. This expression is in accordance with (8).

In the second step we want to obtain (7). Trivially in the expression (8) we can recognize the expressions (5) and (6), thus obtaining (7).

□

From (7) it follows that, if both B_i and B_h have reached the steady state, we can express the span $S_i(\sigma)$ as $S_i(\sigma) = S_i^0 + Q(h,i)$, where

$$Q(h,i) = \max\{0, T_h - H_i\} \tag{9}$$

represents the coupling cost between batches B_h and B_i. For a given solution σ, there is a close relationship between the value of the makespan $T(\sigma)$ and the values $S_i(\sigma)$. Namely, $T(\sigma)$ equals the sum of all the spans $S_i(\sigma)$ plus the tail of the last batch $B_{\sigma(n)}$. In order to express the whole $T(\sigma)$ as the sum of spans, we can regard the tail of the schedule as the span of a dummy batch B_{n+1}, consisting of $b_{n+1} = c+1$ parts having zero processing time on each machine and zero initial and final setup time on each machine. Actually, these processing times can be thought of as arbitrarily small, to indicate that each part of the dummy batch must nonetheless visit all the machines. Notice that $S_{n+1}^0 = 0$, $H_{n+1} = 0$, $T_{n+1} = 0$, and $S_{n+1} = S_{n+1}^0 + Q(\sigma(n), \sigma(n+1))$ so $S_{n+1} = \max\{0, T_n - H_{n+1}\} = T_n$. We can therefore write

$$T(\sigma) = \sum_{i=1}^{n+1} S_i(\sigma) = \sum_{i=1}^{n+1} (S_i{}^0 + \max\{0, T_{\sigma(i)} - H_{\sigma(i+1)}\}) \tag{10}$$

We observe that the problem of finding an optimal sequencing of the batches $B_1, B_2, \ldots, B_n$ is equivalent to finding an optimal cyclic sequencing of the batches $B_1, B_2, \ldots, B_n, B_{n+1}$. In particular, notice that the batch scheduled first (last) in the "open-loop" sequencing is the one following (preceding) B_{n+1} in the cyclic sequencing (notice that, in fact, in the cyclic sequencing the completion time on M_2 of $B_{\sigma(n)}$ coincides with the starting time on M_1 of $B_{\sigma(1)}$). Therefore, from now on we refer to cyclic schedules (in a similar fashion to what done for NO-WAIT FLOW SHOP by Wismer [8]).

As a straightforward consequence of equation (10), the following theorem holds:

Theorem III.2: If, for any sequencing σ, all the batches reach the steady state, BSP is equivalent to a TSP problem in which the coupling cost between batches B_h and B_i is given by equation (9). The value of the makespan $T(\sigma)$ is obtained by adding the optimal solution of the TSP to the sum of the values S_i^0.

From equation (9) we recognize that the structure of this TSP allows the problem to be solved in polynomial time. In fact the cost function becomes identical to that of a two machine NO-WAIT FLOW SHOP. In conclusion, if all the batches reach the steady state in any sequencing σ, the problem can be solved as follows: for each batch B_i, we define a part having processing times H_i and T_i, and apply the Gilmore and Gomory's algorithm [4] to get an optimal sequencing σ in time $O(n \log n)$. The batches will be scheduled according to the optimal sequencing of the corresponding parts.It should be pointed out that this algorithm is guaranteed to yield the optimal solution only if all the batches reach the steady state in any solution σ. In the next section we analyze under what conditions we are ensured that this occurs.

IV. Conditions for the steady state

In the previous section, we have seen that if the batches are "large enough", each of them reaches the steady state, and this enables us to formulate the problem in terms of a special case of TSP. In this section, we compute values b_i^* such that, if $b_i \geq b_i^*$, batch B_i is guaranteed to reach the steady state. In the following, B_h denotes the batch preceding B_i. Let us now turn to the value b_i^*. In general, such a value also depends on the values of the head H_i and tail T_h. However, we next show that the number of parts necessary to reach the steady state never exceeds a figure that only depends on the processing times of batch B_i.

Theorem IV.1: In any sequencing σ, batch B_i reaches the steady state if

$$b_i \geq \left\lceil \frac{c \max(p_{i1}, p_{i2})}{|p_{i2} - p_{i1}|} \right\rceil + 1 \tag{11}$$

Proof: We divide the proof into two subcases, whether $B_i \in N_1$ or $B_i \in N_2$. In both cases, we consider two further subcases, corresponding to the state of the buffer at the end of batch B_h. If batch B_h has reached the steady state, only two cases hold: either the buffer is empty or the buffer is filled up to its capacity.

1. $B_i \in N_1$.

If $p_{i1} < p_{i2}$ the steady state is reached when the buffer is filled up to its capacity. This happens when, for the first time, the part completed on M_1 cannot be moved to the

buffer. The worst case arises when the batch B_h leaves the buffer empty. The final (inital) setup times of batch B_h (B_i) do not affect the number of parts needed to fill up the buffer. We have to find the mininum value of b_i for which $b_i p_{i2} \geq b_i p_{i1} + (c+1)p_{i2}$ holds. That is

$$b_i^* = \left\lceil \frac{cp_{i2}}{p_{i2} - p_{i1}} \right\rceil + 1 \tag{12}$$

The expression (12) is in accordance with (11). If the batch B_h leaves the buffer full or partially full the batch B_i can reach the steady state even sooner.

2. $B_i \in N_2$.

If $p_{i1} > p_{i2}$ the steady state is reached when the buffer is completely emptied. This happens when, for the first time, the part completed on M_1 can be moved to the machine M_2 without wait in the buffer. The worst case arises when the batch B_h leaves the buffer full. In the worst case the final (inital) setup times of batch B_h (B_i) do not affect the number of parts needed to empty the buffer. We have to find the mininum value of b_i for which $b_i p_{i1} \geq b_i p_{i2} + (c+1)p_{i1}$ holds. That is

$$b_i^* = \left\lceil \frac{cp_{i1}}{p_{i1} - p_{i2}} \right\rceil + 1 \tag{13}$$

The expression (13) is in accordance with (11). If the batch B_h leaves the buffer partially full or empty the batch B_i can reach the steady state sooner. The steady state can be reached even sooner than b_i^* parts due to the presence of the setup times of the two batch B_h and B_i.

□

Note the absence of the setup times in the expression (11), this should not be too surprising because the number of parts needed to enter in the steady state in the worst case is not affected by the setup times. In the general case the setup times can help to reach the steady state even sooner because they can help to leave the buffer away from the steady state condition, i.e. empty or filled up to its capacity.

It should be pointed out that the quantity in (11) is not defined for a batch B_q for which $p_{q1} = p_{q2}$. This should not be too surprising because, if the processing times of a batch are very close to each other, the steady state may never be reached: at the end of B_q the value of the difference between the times at which the two machines can start the new batch may be identical to the value immediatly before B_q. In other words, the state of the system may be the same as if B_q had not be scheduled. This would happen expecially if the setup times for each machine are very similar. However, if we consider the batches for which the two processing times are identical as belonging to N_2, the subsequent discussion on the approximation is valid as well.

V. Approximation

In Section IV we have derived an expression for the minimum batch size b_i^* ensuring that batch B_i reaches the steady state. From expression (11) we note that such a value can be very large if the difference between the processing times on the two machines is small. Therefore one is led to analyze the quality of the solution obtained if the same TSP approach is applied to the case in which some batch size b_i is actually lower than b_i^*. In fact, one expects that the closer the batch sizes to the values b_i^*, the more precise the approximation.

In what follows, we derive expressions for an upper and lower bound on the error of a solution, when the batches are not guaranteed to reach the steady state for any sequencing σ. In order to derive such bounds, we analyze how, for a given sequencing σ, the value of the span of a batch B_i is affected by the fact that the preceding batch (let it be B_h) is not in the steady state. In what follows, z^* is the optimal solution to BSP and z_{TSP} denotes the value of the solution obtained by our algorithm. By the previous theorems, if $b_i \geq b_i^*$ for all $i = 1, \ldots, n$, then $z^* = z_{TSP}$. Otherwise, in general this equality does not hold.

In theorem III.1 we have obtained the coupling cost of pairing the batch B_h and B_i when B_h reaches the steady state. If the batch B_h does not reach the steady state then his tail assumes a different value. Let us introduce T_i' as the tail of the batch B_i when the steady state is not reached. In the two following lemmas we derive the expression of T_h' in the cases that $B_h \in N_1$ and $B_h \in N_2$.

Lemma V.1: For any sequencing σ, and for any batch B_i, if $B_h \in N_1$, then $T_h' \leq \max\{0, (c+1)p_{h1} - (b_h - (c+1))(p_{h2} - p_{h1}) + R_{i2} - R_{i1}\}$

Proof: If B_h is in the steady state, the value T_h of the tail is given by (6). If B_h is not in the steady state the buffer is not full, thus T_h' cannot be greater than T_h, whereas in general it can be smaller.

In particular, the most favourable situation arises if, at the time the last part from B_h leaves M_1, the buffer is completely emptied. In this case, $T_h' \leq \max\{0, p_{h2} + R_{h2} - R_{h1}\}$.

Beyond the $(c+1)$-th, each additional part of B_h has the effect of increasing T_h' of the quantity $(p_{h2} - p_{h1})$, and therefore $T_h' \leq \max\{0, p_{h2} + (b_h - (c+1))(p_{h2} - p_{h1}) + R_{h2} - R_{h1}\}$. □

Lemma V.2: For any sequencing σ, and for any batch B_i, if $B_h \in N_2$, then $T_h' \leq \max\{0, (c+1)p_{h2} - (b_h - (c+1))(p_{h1} - p_{h2}) + R_{h2} - R_{h1}\}$

Proof: If B_h is in the steady state, the value T_h of the tail is given by (6). If B_h is not in the steady state the buffer is not empty, thus T_h' cannot be smaller than T_h, whereas in general it can be greater.

In particular, the most unfavourable situation arises if, at the time the last part from B_h leaves M_1, the buffer is filled to its capacity (with parts of type B_h). In this case, $T_h' \leq \max\{0, (c+1)p_{h2} + R_{h2} - R_{h1}\}$.

Beyond the $(c+1)$-th, each additional part of B_h has the effect of decreasing T_h' of the quantity $(p_{h1} - p_{h2})$, and therefore $T_h' \leq \max\{0, (c+1)p_{h2} - (b_h - (c+1))(p_{h1} - p_{h2}) + R_{h2} - R_{h1}\}$. □

Note that for the batches $B_h \in N_1$, and B_h is not steady

state, the value z_{TSP} is an overestimate of the real coupling cost. In the other hand for the batches $B_h \in N_2$, and B_h is not steady state, the TSP cost is an underestimate of the real value z^*. We are now in the position of proving the following theorems that give an upper bound and a lower bound for the z^*.

Theorem V.3: The following upper bound holds:

$$z^* \leq z_{TSP} + \sum_{h \in N_2} (T'_h - T_h) \tag{14}$$

Proof: From Lemma V.2, if B_h does not reach the steady state, z_{TSP} is an underestimate of the real coupling cost z^*.

If $B_h \in N_1$, and B_h not in the steady state, no makespan increases are introduced. If $B_h \in N_2$, and B_h not in the steady state, the increase in the span is smaller than $T'_h - T_h$.

So z^* cannot be greater than $z_{TSP} + \sum_{h \in N_2}(T'_h - T_h)$ and the thesis follows. □

Theorem V.4: The following lower bound holds:

$$z^* \geq z_{TSP} - \sum_{h \in N_1} (T_h - T'_h) \tag{15}$$

Proof: From Lemma V.1, if B_h does not reach the steady state, z_{TSP} is an overestimate of the real coupling cost z^*.

If $B_h \in N_1$, and B_h not in the steady state, the decrease in the span is smaller than $T_h - T'_h$. If $B_h \in N_2$, and B_h not in the steady state, no makespan decreases are introduced.

So z^* cannot be smaller than $z_{TSP} - \sum_{h \in N_1}(T_h - T'_h)$ and the thesis follows. □

From Theorems V.3 and V.4, expressions for the maximum relative error of the solution produced by the algorithm can be given. To this aim, let us observe that a trivial lower bound on z^* is obtained by summing up all the processing and setup times on one machine:

$$z^* \geq \max\{\sum_{h=1}^{n} b_h p_{h1} + U_{h1} + R_{h1}, \sum_{h=1}^{n} b_h p_{h2} + U_{h2} + R_{h2}\} \tag{16}$$

and we can prove the following theorem:

Theorem V.5: The relative error of a solution produced by the algorithm does not exceed

$$\varepsilon \leq \frac{\sum_{h=1}^{n} \max\{0, c \min\{p_{h1}, p_{h2}\} - (b_h - (c+1))|p_{h1} - p_{h2}|\}}{\max\{\sum_{h=1}^{n} b_h p_{h1} + U_{h1} + R_{h1}, \sum_{h=1}^{n} b_h p_{h2} + U_{h2} + R_{h2}\}} \leq \frac{c}{c+1} \tag{17}$$

Proof: If we subtract the value of the lower bound in Theorem V.4 from the upper bound in Theorem V.3, and we divide by the lower bound in (16), we get the first upper bound for ε. The expression of the second term of (17) is maximum when all the batches have minimum cardinality (i.e., $c+1$), when $p_{i1} = p_{i2}$ for all $i = 1, \ldots, n$, and when the setup times $U_{i1}, U_{i2}, R_{i1}, R_{i2}$ are zeros for all $i = 1, \ldots, n$. In this case the bound boils down to the rightmost term in (17). In general, however, the value of this expression is substantially smaller. □

We observed that in the computation of ε, the worst case arises when the setup times are zero. If $U_{i1}, U_{i2}, R_{i1}, R_{i2}$ are non-zeros, for some batch B_i, the value of the rightmost term in (17) is strictly smaller than $\frac{c}{c+1}$. We remind that (17) only gives an upper bound on the relative error, which can in fact be even smaller. Notice that for small c our approach gives a very small relative error, whereas it increases for larger buffer capacity. (This is similar to the approximation result by Papadimitriou and Kanellakis [7].)

VI. Conclusions and future research

In this paper we have dealt with the problem of sequencing batches, with sequence independent setup times, in a flow shop with limited intermediate storage, in order to minimize the makespan. Conditions on the batch sizes were found such that the problem can be solved in polynomial time, by defining a suitable instance of NO-WAIT FLOW SHOP. These conditions are however only sufficient, and a challenging research issue is to refine these conditions. Moreover, the special case in which $p_{i1} = p_{i2}$ has not been addressed explicitly here. The reason is that in this case the subsequent schedule may not be decoupled from the schedule preceding the batch, no matter how large the value b_i.

References

[1] Agnetis, A., Pacciarelli, D., and Rossi, F., *Batch sequencing in a two machine flow shop with limited buffer*, Discrete Applied Mathematics, **72**, 243–260, 1997.

[2] Askin, R.G., Standridge, C.R., *Modeling and analysis of manufacturing systems*, Wiley and Sons, 1993.

[3] Dorn, J., Shams R., *Scheduling high-grade steelmaking*, IEEE Expert. **11**, 1, 28–35, 1996.

[4] Gilmore, P. C., Gomory, R. E., *Sequencing a One State- variable Machine: A Solvable Case of the Traveling Salesman Problem*, Opns. Res. **12**, 655–679, 1964.

[5] Johnson, S. M., *Optimal Two- and Three-Stage Production Schedules with Setup Times Included*, Naval Res. Logistics Quarterly. **1**, 61–68, 1956.

[6] Lixin Tang, Jiyin Liu, Aiying Rong, and Zihou Yang, *A mathematical programming model for scheduling steemaking-continuous casting production*, European Journal of Operational Research. **120**, 423–435, 2000.

[7] Papadimitriou C. and Kanellakis, *Flowshop Scheduling with Limited Temporary Storage*, J. Assoc. Comput. Machinery. **27**, 533–549, 1980.

[8] Wismer, D.A., *Solution of the flowshop scheduling problem with no intermediate queues*, Operations Research. **20**, 689–697, 1972.