DYNAMIC SCHEDULING FOR A SINGLE MACHINE SYSTEM UNDER DIFFERENT SETUP AND BUFFER CAPACITY SCENARIOS

Francesco Martinelli and Paolo Valigi

ABSTRACT

The problem of dynamic scheduling for single machine manufacturing systems has been extensively studied in the past under different setup scenarios, mainly for systems with infinite buffer capacity. This paper addresses a general framework and investigates similarities and differences between policies optimal if setup times and costs are or are not negligible, if buffers have a finite or an infinite capacity. The cost function takes into account of backlog and surplus, but also includes a demand loss component if buffers have a finite capacity and a setup cost if not negligible. Both a steady state and a transient optimization problem are considered and already known results are compared and extended to complete the analysis.

KeyWords: Finite buffers, non negligible setup times and costs, manufacturing systems, scheduling.

I. INTRODUCTION

The scheduling problem for single machine manufacturing systems has been extensively studied in literature, both if setup times are not negligible or if may be neglected. In the first case it is not possible to keep the system with empty buffers at all times and a large literature investigated around interesting stability issues like the existence and stability of limit cycles in the buffer content space ([1-9]). Optimization problems have been defined for this class of systems to minimize a cost function which includes a backlog, a surplus and a setup component. If buffers have a finite capacity also a demand loss component should be included. A large part of literature deals however with infinite capacity systems and the optimization problem mainly consists in finding setup surfaces in the buffer content space, which allow to take decisions like the moment a setup should be performed and which part-type would be more convenient to work on.

In the case of negligible setups, the buffers can be cleared and maintained at all times in the origin of the state space. In this case it becomes of interest to consider a transient optimization problem which, in the case of infinite capacity buffers, has been extensively studied in literature [10-13]. The effect of finite buffers if setup times and costs are negligible has been studied in [14-17].

If setup costs and/or times are not negligible, both the transient and the steady state optimization problems are of interest. Based on a model similar to the one defined in [18], we considered non negligible setup times in [19] and non negligible setup times and costs in [20] deriving results about systems with finite buffers, with particular attention to the transient optimization problem. For this objective, we considered a cost index, similar to the one defined in [21], which allows to optimize the transient behavior. A transient behavior has been considered in [22], where the problem of reaching the limit cycle optimal at steady state in finite time for a two part type one machine manufacturing system characterized by non negligible setup times is considered. We use (as in [19] and in many similar cases [21,23,24]) a dynamic programming approach to derive the structure of the optimal solution.

In this paper we organize and extend all this material and investigate around differences and similarities between the different scenarios. In particular, it will be remarked that under all possible setup situations, the finite buffers always imply, if the demand loss cost is
small enough, a kind of inversion in the optimal solution of the infinite capacity case, as it will be illustrated below. This was initially observed for systems with negligible setups in [15].

Other papers on related subjects include [25,26], and [27]. In [25] a stochastic model is considered for a setup optimization problem very similar to the one addressed in this paper, although the optimization does not include buffer holding costs. In [26] an analytical characterization of necessary conditions for optimal setup changes is given, while in [27] the problem of finding the optimal scheduling under different production constraints is considered.

The paper is organized as follows. In Section 2 we introduce some notation and provide the problem formulation. In Section 3 we address the optimization problem for a system with negligible setup times and costs, in Section 4 we study the case of non-negligible setup times and negligible costs, in Section 5 the case in which both setup times and costs are not negligible, and in Section 6 the case in which setup times are negligible while costs are not. Section 7 concludes the paper.

II. NOTATION AND PROBLEM FORMULATION

We consider the manufacturing system depicted in Fig. 1, comprising a single machine producing two part types. A fluid approximation is considered. Let \( d_i \) be the constant demand rate and \( \mu_i \) be the maximum production rate for part type \( i, i = 1, 2 \), and let \( L_i \) be the capacity of buffer \( i, i = 1, 2 \), either infinite or finite. Let \( x(t) = [x_1(t), x_2(t)]^T \) be the system state, with \( x_i(t) \) being the content at time \( t \) of buffer \( i, i = 1, 2 \), and let the state space be denoted as \( \mathcal{X} = \{ x \in \mathbb{R}^2 : 0 \leq x_i \leq L_i, i = 1, 2 \} \).

According to Gershwin’s hierarchical structure [28,29], demand can be modeled as a constant continuous flow at the level of setup scheduling. If \( u(t) \) is the actual production rate at time \( t, 0 \leq u(t) \leq \mu_i \) if producing part type \( i \), the dynamic evolution for part type \( i \), if \( 0 < x_i(t) < L_i, i = 1, 2 \), is described by

\[
\dot{x}_i = d_i - s_i(t)u(t),
\]

with initial conditions \( x(0) = x_0 \) for both part types, where \( s(t) = [1, 0]' \) if the machine setup is 1, \( s(t) = [0, 1]' \) if the machine setup is 2, and \( s(t) = [0, 0]' \) during a setup change.

In any manufacturing system, switching production among different part-types implies a setup process, which can be described in terms of time and cost required to carry out the reconfiguration of the machine. Depending on the system, setup times and costs may be negligible or not negligible. The main contribution of this paper is the study of the impact of these parameters on the optimal control of the system and in highlighting some similarities in the different scenarios.

Let \( \tau_j \) and \( K_j \) be the setup time and cost respectively to change production from part type \( i \) to part type \( j \neq i \). Consider the following cost index:

\[
J_s = \lim_{r \to \infty} \frac{1}{r} \int_0^r \left[ g_x(x(t)) + K_{12}n_{12}(t) + K_{21}n_{21}(t) \right] dt,
\]

(2)

where \( g(x) \) is a positive instantaneous cost function associated with buffer content \( x \) and \( n_j(t) = 1/\tau_j \) during a setup from part type \( i \) to part type \( j \) and \( n_j(t) = 0 \) otherwise. The integral in (2) must be computed in the framework of the generalized functions to include the case of negligible setup times \( (\tau_j \simeq 0) \). The trajectory minimizing \( J_s \) is the steady-state optimal trajectory.

Once the steady state optimal trajectory is known, it is interesting to study the best way in some sense of reaching it. This is a transient optimization problem, particularly interesting if the machine is failure prone and the optimal steady state trajectory must be periodically recovered. In this case, we will assume that failures rarely occur, allowing the system to recover the steady state situation after each failure.

Since the index \( J_s \) in (2) is independent of the transient behavior of the system, we consider for the transient optimization problem the following cost index (see also [21]):

\[
J_t = \lim_{r \to \infty} \inf \left[ \int_0^r \left[ g_x(x(t)) + K_{12}n_{12}(t) + K_{21}n_{21}(t) - J \right] dt \right],
\]

(3)

where \( J \) is the average cost of the limit cycle at steady state which optimizes \( J_s \) and will be derived in the following sections.

Let \( R_i \) be the cost for the unit of lost demand of part type \( i, i = 1, 2 \). Then, the holding cost function considered in this paper is:

\[
g(x) = c_i x_1 + c_2 x_2 + R_i d_i I(x_1 = L_i) + R_2 d_2 I(x_2 = L_2),
\]

(4)

where \( I(\cdot) = 1 \) if the argument is true and 0 otherwise.

Throughout the paper, the following stability assumption will be assumed, although not explicitly men-
tioned. For this reason it will be labeled as Assumption 0.

**Assumption 0.** Production rates are large enough to assure existence of a convergent limit cycle, i.e.:

\[
\frac{d_1}{\mu_1} + \frac{d_2}{\mu_2} < 1
\]

In the following, we will study the problem of optimizing the steady-state cost index \(J_o\) in (2) and the transient cost index \(J_t\) in (3), under different classes of setup process, corresponding to different values of the setup times \(\tau_{ij}\) and costs \(K_{ij}, i = 1, 2, j = 2, 1\).

### III. OPTIMIZATION PROBLEM IN THE CASE SETUP TIMES AND COSTS ARE NEGligible

In this section we address the optimization problem introduced in Section 2 if the production system is flexible and both setup times and costs may be considered negligible. This kind of approximation is frequent in modern flexible manufacturing systems where machine tool change and/or machine reconfiguration are characterized by times and costs negligible if compared with typical part processing times.

#### 3.1 The steady state case

It is straightforward to verify that both the buffers of the system considered in this paper may be cleared in finite time if setup times are negligible. If, in addition, setup costs are also negligible, it is possible to keep the system with empty buffers without incurring in any cost. So, the optimal solution in this case is given by any control policy which clears the buffers in finite time and then keeps them at the origin for all subsequent times. This trivially gives for the cost index \(J_o\) a value \(\mathcal{J} = 0\) which is then the optimal value.

#### 3.2 The transient optimization problem

While the steady state optimization problem is trivial in the case of negligible setup times and costs, the transient optimization problem has been extensively studied in the past both if buffers have a finite or an infinite capacity.

Since in this case, as observed above, the steady state optimal value for \(J_t\) is \(\mathcal{J} = 0\), \(J_t\) in (3) is given by the following expression:

\[
J_t = \liminf_{T \to \infty} J_t^T g(x(t))dt.
\]

If buffers have an infinite capacity, index \(J_t\) in (6) is optimized by the well known \(c_t\mu\)-rule [10-12], or, more in general, by a myopic policy [13]. Observe that, if the system is symmetric (see subsequent Assumption 1), in the case of infinite buffers any full capacity policy (i.e., a policy working at full capacity, that is such that \(\sum u_i/\mu_i = 1\) until the buffers are empty and then keeps them in the origin) is optimal and clears the buffers in minimum time.

If buffers have a finite capacity, the \(c_t\mu\)-rule is no longer optimal (see [17]). It could be convenient to exploit in this case the faster velocity which can be attained along the borders of the state space (i.e. on the lines \(L_1 := \{ (x_1, x_2) \in \mathbb{R}^2 : x_1 = L_1, 0 \leq x_2 \leq L_2 \}\) and \(L_2 := \{ (x_1, x_2) \in \mathbb{R}^2 : x_2 = L_2, 0 \leq x_1 \leq L_1 \}\)) to reach the origin of the state space with a smaller cost (see [17] for details). Two main features may be observed in this case:

- In a region far from the origin of the state space, the optimal policy produces the part type with minor backlog (i.e. it is a Shortest Queue First (SQF) policy). See [15] for details.
- There is a point \(P_i\) on each border \(L_i\) of the state space, whose position \((L_i, X_i)\) if \(i = 1\) or \((X_i, L_i)\) if \(i = 2\) depends on parameters of the system (but not on initial conditions), such that if the optimal trajectory is moving along \(L_i\) it will lose customers of part type \(i\) if \(X_i > X_i\) and will avoid to lose demand otherwise. Please refer to [17] for details.

The features highlighted above will be also confirmed in the case of non negligible setup times and costs, as it will be remarked below.

### IV. THE OPTIMAL CONTROL PROBLEM IN THE CASE OF NON-NEGLIGIBLE SETUP TIMES AND NEGLIGIBLE SETUP COSTS

In this section we will study the general problem introduced in Section 2, for the case of non-negligible setup times and negligible setup costs, corresponding to the following parameters for the setup process:

\[
\tau_{1,2} > 0, \quad \tau_{2,1} > 0, \quad K_{1,2} = 0, \quad K_{2,1} = 0,
\]

and to the following structure for the cost indices:

\[
J_o = \lim_{T \to \infty} \frac{1}{T} \int_0^T [g(x(t))]dt,
\]

\[
J_t = \liminf_{T \to \infty} \int_0^T [g(x(t)) - \mathcal{J}]dt,
\]

In addition, we will study the problem under the simplifying assumption that the system is symmetric.

**Assumption 1.** The system is symmetric, that is, \(\mu_1 = \mu_2 = \mu, d_1 = d_2 = d, c_1 = c_2 = c, \tau_{1,2} = \tau_{2,1} = \tau, L_1 = L_2 = L, R_1 = R_2 = R\).
4.1 The steady-state case

4.1.1 Infinite buffer capacity

The trajectory minimizing $J_a$ under conditions (7) on setup process parameters and assuming infinite capacity, i.e., $L = \infty$, is the limit cycle $C$ closest to the origin [18], as reported in Fig. 2, and located as in the figure for a system with parameters $\mu_1 = 3, d_1 = d_2 = 1, \tau = 5$ and $g(x) = c(x_1 + x_2)$, with any positive $c$.

In the limit cycle, at point $(, 0)$ the system performs a setup from part type 2 to part type 1 and completes it at point $(, )$, where a production cycle on part 1 is started. It is easy to compute the cost $J$ of the optimal limit cycle:

$$J = \frac{2c\tau d(\mu - d)}{\mu - 2d} \tag{10}$$

4.1.2 Finite buffer capacity

We will now examine the steady-state solution in the case of finite buffer capacity, i.e., $L < \infty$, under the assumption that $L$ is large enough that the state space wholly contains the limit cycle $C$. This is reasonable, since a proper choice of the buffer capacities should at least take into account optimal steady state behavior.

Assumption 2. The optimal limit cycle $C$ never reaches the bound of the state space $X$, i.e., $\frac{2d\tau (\mu - d)}{\mu - 2d} < L$.

Assumption 3 makes the problem non-trivial.

Assumption 3. The cost of maintaining the system in the corner $(L, L)$ is larger than the average cost on the limit cycle $C$.

Assumption 3 holds if the following conditions are satisfied, for the case of non-symmetric and symmetric systems, respectively:

$$c_1 L_1 + c_2 L_2 + R_1 d_1 + R_2 d_2 > J \tag{11}$$

$$2(cL + Rd) > J \tag{12}$$

where $J$ in this case is given by (10). Under Assumptions 2 and 3, the optimal trajectory solving the steady-state problem for finite buffers is the limit cycle $C$ closest to the origin [18], as in the case of infinite buffer capacity (see Fig. 2), hence the optimal value $\bar{J}$ for $J_a$ is again given by (10).

4.2 The transient problem

The transient optimization problem is that of bringing the system back to the limit cycle upon machine failure, or similar reasons. To this respect, the cost index $J_t$ given in (9) (with $J$ given by (10)) is used, whose minimization implies a transient behavior with minimum cost, i.e., a way to reach the limit cycle with minimum cost. It is recalled that the $\lim\inf$ operation is needed since the integral reaches a periodic steady state behavior.

The main objective of this Section is to optimize $J_t$, i.e., to solve the optimal transient problem for the infinite buffer case and for the finite buffer one. In both situations, the analysis will only cover the case the initial state is a point reached after a failure occurred while moving on the limit cycle, as stated in the subsequent Assumption 4.

Assumption 4. Let $x_0 = (x_{01}, x_{02})$ be the initial state considered for the transient problem. Then, there exists a point $P = (x_1, x_2)$ lying on the limit cycle $C$ and a time $\Delta t$ such that $x_{0i} = x_i + d_i \cdot \Delta t, i = 1, 2$.

In both the finite and infinite cases, it will be shown that the optimal policy works at full capacity (i.e. $u(t) = \mu_i$ if producing part type $i$). So, the optimization problem is that of finding the initial setup and the setup curves. The problem will be studied under the Assumptions 1 through 4.

4.2.1 Infinite buffer capacity

Now we present some analytical results for the infinite buffer capacity case. They give insight into the structure of the optimal solution and represent prelimi-
nary material for the finite buffer case as well. In the infinite case the instantaneous cost function is given by:

\[ g(x) = c_1 x_1 + c_2 x_2 = c(x_1 + x_2), \]  

(13)

where \( c_1 \) and \( c_2 \) are positive constants. The following theorems characterize the optimal solution. These results are in accordance with the structure of the optimal policy for the finite capacity case found through dynamic programming, as it will be remarked in the next section.

It turns out that the optimal policy always produces at maximum allowed capacity (Theorem 1), performs setups on the axis of the state space (Theorem 2), and the initial setup is selected maximizing the time to the next setup, i.e., using a LQF (Longest Queue First) policy (Theorem 3).

**Theorem 1.** For a given initial state \( x_0 \), under Assumption 1 and 4, let \( u(t) \), \( t \in (0, \infty) \), be the control minimizing \( J_r \). Then, if \( s(t_k) = 1 \), \( u(t_k) = \mu \) whenever \( x(t_k) > 0 \), for all \( t \geq 0 \).

**Proof.** Without loss of generality, assume there is a time interval \( (t_1, t_2) \), \( t_2 > t_1 \), where \( x(t) > 0 \), \( s(t_1) = 1 \) and \( u(t) = \mu \) with \( u < \mu \). It is easy to construct another control function \( \bar{u}(t) = \mu \) for \( t \in (t_1, t_1 + \frac{u - d}{\mu - d}(t_2 - t_1)) \) and \( \bar{u}(t) = d \) for \( t \in (t_1 + \frac{u - d}{\mu - d}(t_2 - t_1), t_2) \), giving a smaller value for \( J_r \).

**Theorem 2.** Assume \( x(t) \) is an optimal trajectory. Let \( x(\bar{T}) = (\bar{x}_i, \bar{x}_j) \) be a point of the trajectory \( x(t) \) where a setup is performed. Then, either \( \bar{x}_i = 0 \) or \( \bar{x}_j = 0 \), that is, the setup is performed on the axis \( x_2 = 0 \) or \( x_1 = 0 \), respectively.

**Proof.** Assume, without loss of generality, the optimal trajectory \( x(t) \) performs a setup from part type 1 to part type 2 at time \( \bar{T} \) in \( R_0 = (a, b) \), \( a \neq 0, b \neq 0 \). Consider a trajectory \( \tilde{x}(t) \) coincident with \( x(t) \) up to time \( \bar{T} \) and let it perform the setup from part type 1 to part type 2 at time \( \bar{T} + \Delta \tau \). Consider \( R_0 = (a - (\mu - d)\Delta \tau, b + d\Delta \tau) \), with \( a \geq (\mu - d)\Delta \tau \). The remaining part of the trajectory \( \tilde{x}(t) \) is realized by performing the \( i \)-th setup from part type \( i \) to part type \( j \) at time \( \tilde{t}_m^{(i)} \), where \( \tilde{x}_i(t_m^{(i)}) = x_i(t_m^{(i)}) \) and \( t_m^{(i)} \) is the time the optimal trajectory performs the same setup. As it is possible to verify, \( \tilde{t}_m^{(i)} > t_m^{(i)} \) for all \( i, j, m \), thus achieving a lower cost \( J_f \) for \( \tilde{x}(t) \).

**Theorem 3.** Let \( x_0 = (a, b) \) be the initial state at time \( t = 0 \), and assume \( a > b \) \( (a < b) \). Then the optimal initial setup is \( s(0) = (1, 0) \) \( (s(0) = (0, 1)) \).

**Proof.** Assume \( x_0 = (a, b) \) with \( a > b \). The proof is based on the comparison between the cost of the trajectory \( T_1 \) starting from \( x_0 \) with setup 1 and trajectory \( T_2 \) starting from \( x_0 \) with setup 2. We know from Theorem 2 that both the trajectories, in order to be optimal, need to perform setups on the axis. Let \( \alpha_0 \) be the coordinate different from 0 of the \( k \)-th setup point of trajectory \( T_2 \) and \( \beta_k \) of \( T_1 \). Let \( \gamma_k = \beta_k - \alpha_k \) and \( \Delta \tau = c(t^{(1)}_k - t^{(2)}_k) \), where \( t^{(1)}_k \) is the time the trajectory \( T_1 \) performs the \( k \)-th setup and \( t^{(2)}_k \) is the time the trajectory \( T_2 \) performs the \( k \)-th setup. The corresponding dynamics are:

\[ \Delta \tau_{k+1} = \Delta \tau_k + \gamma_k \frac{\gamma_k}{\mu - d}, \quad \gamma_{k+1} = \frac{d}{\mu - d} \gamma_k, \]

with initial conditions \( \Delta \tau_0 = \frac{a - b}{\mu - d} \), \( \gamma_0 = \frac{\frac{a - b}{\mu - d} \Delta \tau_0}{\mu - d} \).

Since \( \frac{d}{\mu - d} < 1 \), \( \gamma_k \to 0 \) as \( k \to \infty \), and \( \Delta \tau_k \to 0 \) as \( k \to \infty \). In addition, \( \Delta \tau \) reaches 0 monotonically. This implies the cost \( J_f \) of \( T_2 \) is larger than the cost of \( T_1 \).

### 4.2.2 Finite buffer capacity

Let us now assume finite buffer capacity, given by \( L < \infty \). Let \( R \) be the cost for the unit of lost demand of the two part types, then, we use the following instantaneous cost function:

\[ g(x) = c(x_1 + x_2) + Rd[I(x_1 = L) + I(x_2 = L)], \]

(14)

and the following transient cost index

\[ J_f = \lim \inf \int_0^T \left( g(x(t)) - \bar{J} \right) dt, \]

(15)

obtained by (4) and (3), respectively, and \( \bar{J} \) is the cost of the limit cycle \( C \) given by (10).

To derive the structure of the optimal policy, we used an approximate dynamic programming approach. It is possible to apply the following dynamic programming approach since Theorem 1 also holds in the finite buffer case, as it is possible to prove following an argument similar to the one used to prove the theorem in the infinite buffer capacity case.

To use dynamic programming, we introduce a spatial and temporal discretization. Let \( \Delta \tau \) be the time quantum, selected in such a way that the setup time is a multiple of it: \( \tau = N \Delta \tau \). Then, let \( J^i(x) \) denote the optimal cost-to-go function (i.e. the optimal value for the integral in the cost function (15) before performing the \( \lim \inf \) operation) starting from \( x \) with setup \( i \) \((i = 1, 2) \) and moving for a time \( k * \Delta \tau \).

Let
denote the cost of the trajectory which does not perform a setup in state \( x \) and continues working part type \( i \) at least for the first step (i.e. from time 0 to \( \delta t \)). Let

\[
J_W = \left( \int_0^\infty g(x+u, \tau) d\tau + J_i^{(k-1)}(x+u, \delta t) + (k-1)\tau \delta t \right) - k\tau \delta t
\]

\[
J_S = \left( \int_0^\infty g(x+u, \tau) d\tau + J_i^{(k-N)}(x+u, \tau) + (k-N)\tau \delta t \right) - k\tau \delta t
\]

denote the cost if a setup is immediately performed in state \( x \). The dynamic programming equations, for \( k > N \), \( i = 1, 2 \) are then given by:

\[
J_i^{(k)}(x) = \min \{ J_W, J_S \},
\]

where \( j \neq i \) is the other part type, \( u_0 = (d, d), u_1 = (d - \mu, d), u_2 = (d, d - \mu) \). Of course, the cases of \( x \) on the border of the state space have been properly handled, as well as the first \( N \) iterations of the equations in (16); in particular, we set \( J_i^{(0)}(x) = 0 \) for all \( x \) and \( i, i = 1, 2 \). Iterating equations (16), does not converge to a limit value. Rather, at steady state, the cost-to-go functions oscillate in a small range.

In the following we describe the specific behavior observed with finite buffers, derived through the dynamic programming approach as \( R \) changes. Some analytical results are also included.

**Result 1.** If the demand loss cost \( R \) is large enough, the optimal control may avoid to reach the border \((x_1 = L \text{ or } x_2 = L)\) of the state space performing the setup before reaching the border. If \( R \) is reduced, the optimal control will chose to move along the border up to a given point, depending on the value of \( R \). If \( R \) is small enough, the trajectory will cover all of the border performing the setup on the axis, i.e. at point \((0, L)\) or \((L, 0)\).

Result 1 is illustrated in Figs. 3 and 4, for a system with parameters \( \mu = 3, d = 1, L = 62, \tau = 5 \), and initial conditions \( x_0 = (50, 50) \). Figure 3, up, refers to a very large value of \( R \), \( R = 10^6 \); the optimal trajectory totally avoids the border. As \( R \) decreases, the trajectory starts to move along the border: Fig. 3, down, is obtained with \( R = 1000 \), Fig. 4, up, with \( R = 600 \) and Fig. 4, down, with \( R = 150 \). The reason for such a behavior is that moving along the border gives a reduction rate in holding cost of \( c(\mu - d) \) larger than the rate \( c(\mu - 2d) \) corresponding to an internal trajectory. If such a contribution is larger than the one corresponding to demand loss, the trajectory will move along the border.

**Result 2.** There is a point \( P \) on the border \( x_2 = L \), with \( P = (X_1, L) \), \( X_1 \) depending on the parameters, such that, if initial setup is \( s(0) = (1, 0) \), the optimal trajectory moves along the border \((x_1, L) \), \( 0 \leq x_1 \leq L_1 \), if \( x_1 > X_1 \), and per-
forms a setup if \( x_1 < X_1 \). A similar property holds for the border \( x_1 = L \).

This result is illustrated in Fig. 5 for \( R = 1000 \). The point \( P \) does not depend on initial conditions. This property is similar to the one found for systems with negligible setup times and costs (see item 2, Section 3.2).

**Result 3.** Assume \( R \) is small enough (hence, setup changes occur on the axes, Result 1). Then, for each point \( x_0 = (a, b) \) there is a threshold \( R_t(x_0) \), such that: a) if \( R > R_t(x_0) \), the optimal initial setup from \( x_0 \) is performed according to a LQF policy as in the infinite capacity case; b) if \( R < R_t(x_0) \) the optimal initial setup is performed according to a SQF (Shortest Queue First) policy. Since \( R_t(x_0) \) is constant on lines \( a + b = \text{constant} \), as shown below, we observe a kind of inversion in the initial setup (see Fig. 6).

Result 3 is illustrated in Fig. 6, with the same choice of parameters considered in the figures above, with \( R = 50 \) in Fig. 6, up, and \( R = 150 \) in Fig. 6, down. A similar property can be observed also in the case of negligible setup times and costs (see item 1, Section 3.2).

**Proof.** Assumption 2 implies that once the trajectory leaves the border on the axis, then it will never reach again any border. The expression of the threshold is obtained through a direct comparison between the cost \( J_t \) of the trajectory starting from \( x_0 \) with setup 1 and the trajectory starting from \( x_0 \) with setup 2. (See Fig. 7). This comparison gives:

\[
R_t(x_0) = \frac{\Delta A - J_0}{d(t_{01} - t_{02} - t_{2s})},
\]

where, according to the figure, \( t_{01} \) is the time the LQF trajectory reaches the border, \( t_0 \) the time the SQF trajectory reaches the border, \( t_{01} + t_{1s} \) the time the LQF trajectory reaches the axis and \( t_{02} + t_{2s} \) the time the SQF trajectory reaches the axis, \( J \) the average cost on the limit cycle, \( \Delta A \) is the shaded area represented in the figure. Expanding this, gives the expression (17) of \( R_t(x_0) \).

From the expression of \( R_t(x_0) \), it is clear that initial states \( x_0 = (a, b) \) with the same value of \( a + b \), have the same threshold \( R_t(x_0) \), according to the inversion of Fig. 6. As \( a + b \) increases (with maximum on the corner of the state space \((L, L)\), from which it is possible to stay on the border for more time), \( R_t(x_0) \) also increases: the inversion starts from the corner of \( X \) and propagates in the interior as \( R \) decreases. This is clear from Fig. 6.
V. THE OPTIMAL CONTROL PROBLEM IN THE CASE OF NON-NEGligible SETUP TIMES AND COSTS

In this section we will study the general problem introduced in Section 2, for the case of non-negligible setup times and costs:

\[ \tau_{1,2} > 0, \quad \tau_{2,1} > 0, \quad K_{1,2} > 0, \quad K_{2,1} > 0, \quad (18) \]

under the simplifying Assumption 1 that the system is symmetric, already introduced in Section 4.

5.1 The steady-state case

In this section we consider the problem of determining the optimal trajectory in the steady state case for a system with non-negligible setup times and costs, i.e. the trajectory minimizing the index \( J_a \) in (2). Due to the setup costs, we expect a steady state optimal trajectory different from the limit cycle optimal under negligible setup costs (see Fig. 2). As a matter of facts, to minimize the backlog costs, the setup frequency should be reduced following a trajectory as far as possible from the origin.

5.1.1 Infinite buffer capacity

To clarify the matter, we have tackled the problem through a dynamic programming approach. Using the notation introduced in the previous section, the dynamic programming equations, for \( k > N \), and denoting by \( i, j = 1, 2 \), the part-type under production, are given by:

\[
J_i^{(k)}(x) = \frac{1}{k \delta t} \min \{ J_w, J_s \},
\]

\[
J_w = \int_0^t g(x + \nu_1 t) dt + (k-1) \delta J_i^{(k-1)}(x + \nu_1 \delta t),
\]

\[
J_s = \int_0^t g(x + \nu_2 t) dt + (k-N) \delta J_j^{(k-N)}(x + \nu_2 \tau) + K_s,
\]

where \( j \neq i \) is the other part type, \( \nu_1 = (d, d), \nu_1 = (d - \mu, d), \nu_2 = (d, d - \mu) \), \( J_w \) is the cost if the system keeps working on part-type \( i \) for an additional time step, \( J_s \) is the cost if a setup is triggered from part-type \( i \) to \( j \). As in the previous section, the cases of \( x \) on the border of the state space have been properly treated, as well as the \( N \) first iterations of the equations in (19); in particular, we set \( J^{(0)}(x) = 0 \) for all \( x \) and \( i = 1, 2 \). Iterating equations (19), the cost-to-go function reaches a steady state value, as \( k \) increases.

In Fig. 8 we give the optimal trajectory at steady state obtained using the dynamic programming approach (i.e. equation (19)) on a system with parameters \( d = 1, \mu = 3, \tau = 5, c = 3, R = 100, L = 62 \) and \( K_s = 2000 \). From this and other examples, we have observed that the optimal trajectory at steady state presents the shape of the limit cycle represented in the figure: from point \( A = (x_A, 0) \) the system, with setup 2, meets the demand of part type 2 at rate \( d \) (it can not produce at a faster rate, since \( x_2 = 0 \)) up to point \( B = (x_B, 0) \). The parameter \( l = x_B - x_A \) characterizes the limit cycle and its optimal value will be analytically derived next. After point \( B \), the system performs a setup reaching \( C = (x_C, y_C) \), with \( x_C = x_B + d \cdot \tau \) and \( y_C = d \cdot \tau \) incurring in a setup cost \( K_s \). In \( C \) the machine is ready to work on part type 1: with a full production rate \( \mu_1 = \mu_1 \), the trajectory reaches \( D = (0, y_B) \) at \( (0, x_A) \). From this point, the trajectory is symmetrically equivalent to the part of the limit cycle just described. The notation \( \mathcal{C}(l) \) will denote a limit cycle with the shape reported in Fig. 8, with \( x_B - x_A = d \cdot l \).

The following Lemma provides the value \( l^* \) corresponding to the limit cycle \( \mathcal{C}(l^*) \) achieving the minimum cost \( J_\mu \) among all possible limit cycles \( \mathcal{C}(l), \ l \geq 0 \). Based on the observation of the dynamic programming results, the set \( \mathcal{C} = \{ \mathcal{C}(l), \ l \geq 0 \} \) is expected to contain the optimal solution to the general problem of minimizing \( J_\mu \), considering a set of system parameter values which should at least satisfy Assumptions 1, 2 and 3.

**Lemma 1.** Assume \( L \) is infinite and let
\[ l_0 = \frac{-cd\mu^2 + \sqrt{cd\mu(\mu - 2d)^2[2K_s(\mu - d) + cd\mu\tau^2]}}{cd\mu(\mu - d)}. \] (20)

Then, the problem of finding
\[ l^* = \arg\min_{l \in (0,100)} J_a \] (21)
is solved by
\[ l^* = \max\{l_0, 0\}. \] (22)

**Proof.** The proof is based on the direct computation of \( J_a \) on a generic limit cycle \( C(l) \). Then, let \( D(l) \) be the derivative of such a cost with respect to \( l \). The function \( D(l) \) is definitely positive as \( l \) goes to \( \infty \) and \( l_0 \) given in (20) is the largest solution of \( D(l) = 0 \), and it is also the only solution which can be positive. 

Equations (20) and (22) agree with intuition: \( l^* \) is a function of the setup cost \( K_s \) going to infinity as \( K_s \) goes to infinity. Also, if \( K_s = 0 \), we expect \( l^* = 0 \), since this is the optimal solution of the case with negligible setup costs. In fact, according to Lemma 1, setting \( K_s = 0 \) in \( l_0 \), we get a negative value, hence \( l^* = 0 \). It is interesting to observe that \( C(l) \), i.e. the limit cycle with \( l = 0 \), which is optimal in the case setup costs are negligible (see Fig. 2), remains optimal also for some positive \( K_s \in (0, K_s^*) \), with a proper \( K_s^* > 0 \). The following lemma, which can be proved through a direct computation, gives the value of this \( K_s^* \).

**Lemma 2.** Consider \( l^* \) given in Lemma 1 as a function of the setup cost \( K_s \). Then, \( l^*(K_s) = 0 \) for all \( K_s \in (0, K_s^*) \), with
\[ K_s^* = \frac{2c\mu d^2\tau^2}{(\mu - 2d)^2}. \] (23)

As an illustration of the results above, we present some numerical examples, obtained again through the dynamic programming approach applied to the system considered above, i.e. the system with parameters \( d = 1, \mu = 3, \tau = 5, c = 3, L = 62, R = 100 \) and \( K_s \) varying. With these parameters, equation (23) gives \( K_s^* = 450 \). In Figs. 8 and 9 we report the limit cycle for \( K_s = 200, 500 \) and 350 respectively. From equations (20) and (22) we get \( l^* = 7.6, 0.36 \) and 0 respectively. These values agree with the values which can be observed in the figures, in particular the optimal steady state trajectory approaches the limit cycle optimal in the case of negligible setup costs (see Fig. 2) as \( K_s \) becomes smaller than \( K_s^* \).

### 5.1.2 Finite buffer capacity

We will now examine the steady-state solution in the case of finite buffer capacity, i.e., \( L < \infty \), under Assumption 2 and 3 already introduced in Section 3. Under these assumptions, numerical evidence indicates that the optimal trajectory solving the steady-state problem for finite buffers is the limit cycle \( C(l^*) \), as in the case of infinite buffer capacity (see Fig. 8).

### 5.2 The transient problem

We now consider the transient optimization problem for the case the setup times and costs are not negligible. So, the index cost to be optimized is \( J_t \) in (3), with \( \overline{J} \) being the average cost (including setup costs) on the optimal limit cycle \( C(l^*) \) determined above. The procedure is as follows: given a system, we determine the average cost on the limit cycle \( C(l^*) \), denote this cost as \( J \) and solve a dynamic programming optimization problem (see below) to determine the trajectory which minimizes \( J_t \) starting from a given initial state \( x_0 \) and a given initial setup \( i, i = 1, 2 \). We also discuss the case the initial setup can be selected and find a result similar to the one offered in the previous section for the negligible setup cost case. As already remarked, observe that the integral in (3) is finite since at steady state, on the limit cycle, the term \( \int_{0}^{T} (g[x(t)] + K_s[n_{12}(t) + n_{13}(t)])dt \) reaches infinity as \( \overline{J} \).

The cost index \( J_t \) depends on the transient behavior of the system. Minimizing \( J_t \) implies a transient behavior with minimum cost, i.e. a way to reach the limit cycle...
with minimum cost. Finally, the \( \lim \inf \) operation is needed since the integral reaches a periodic steady state behavior. In this case, the dynamic programming equations referred to the optimization of \( J_t \), are given by:

\[
J^{(k)}_t(x) = \min \{ J^{(k)}_w, J^{(k)}_s \}, \\
J^{(k)}_w = \int_0^t g(x + v_t) dt + J^{(k-1)}_i(x + v_t \delta t) - \bar{J} \delta t, \\
J^{(k)}_s = \int_0^t g(x + v_t) dt + J^{(k-1)}_i(x + v_t \bar{\tau}) + K_s - N \bar{J} \delta t
\]

Applying this equation to the same system considered above, i.e. the system with parameters \( d = 1, \mu = 3, \tau = 5, K_s = 2000, L = 62, c = 3 \) and \( R \) varying, we obtain the trajectories of Figs. 10 and 11, respectively corresponding to \( R = 100, 1000 \) and \( 100000 \). As expected, according to Assumptions 2 and 3, the limit cycle is independent of \( R \) which only characterizes the way the limit cycle is reached, i.e. the transient optimal trajectory. As it can be observed, as \( R \) increases, the trajectory ends to totally avoid the border of the state space, where a penalty for demand loss is incurred. In addition, as already observed, if the initial setup is a control variable, comparing the values of the two cost-to-go functions \( J_i \) and \( J_2 \) (corresponding to initial setup \( i, i = 1, 2 \)) from the same initial state \( x_0 = (a, b) \), it is possible to decide which is the part type to start with. In Fig. 12 we report the optimal initial setup selection for two different values of \( R, R = 100 \) and \( R = 200 \), respectively. As reported in the previous section, it is possible to provide an expression for the threshold \( R_L(x_0) \) such that, if \( R > R_L(x_0) \), the optimal initial setup from \( x_0 \) obeys a LQF (Longest Queue First) policy while, if \( R < R_L(x_0) \), the optimal initial setup implements a SQF (Shortest Queue First) policy. Since \( R_L(x_0) \) is constant on lines \( a + b = \text{constant} \), as shown below, we observe a kind of inversion in the initial setup (see Fig. 12).

The following theorem explicitly states this property and provides the expression of the threshold \( R_L(x_0) \). Its proof is as in Theorem 4, considering now the new
value of \( \overline{J} \), which is the average cost on the optimal limit cycle \( C(\hat{r}) \).

**Theorem 5.** Assume \( R \) is small enough that, starting from \( x_0 = (a, b) \), the optimal trajectory travels on the border up to the axis (like in Fig. 10). Then, under Assumptions 1, 2 and 3, there is a threshold \( R(x_0) \) such that, if \( R < R(x_0) \), it is beneficial to start from \( x_0 \) with setup 1 if \( a < b \), otherwise, if \( R > R(x_0) \), with setup 2. The expression of this threshold is:

\[
R_1(x_0) = \begin{cases} 
R_{i1} & \text{if } l_0 > 0 \\
R_{i2} & \text{if } l_0 \leq 0 
\end{cases} \tag{25}
\]

where \( l_0 \) has been defined in equation (20), \( R_{i1} \) is given by:

\[
R_{i1} = [-SR(2L-a-b)\mu^2 - SR(a+b)d\mu + 4\mu^2 d^2 \tau^2 + 6\mu d SRLd \mu - 2\mu^3 d^2 \tau \tau^2 + 12\mu^2 d^3 \tau \tau \dot{\tau} - 8\mu d^4 \tau \dot{\tau} - 2\mu d^2 \tau \dot{\dot{\tau}} \tau - 4\mu^3 d^2 \tau \tau^2 \tau]c \\
/[2d(\mu - 2d)SR(\mu - d)],
\]

with \( SR = \sqrt{\mu d(\mu - 2d)^2 (d \mu \tau^2 + 2K_s(\mu - d))} \) and, finally, \( R_{i2} \) by:

\[
R_{i2} = [\mu^3 \tau (a + b - 2L) - 2 \mu^2 \tau (a + b)d - 8 \mu^3 \tau K_s + 8 \mu^2 \tau L \dot{d} + 8 \mu d^2 K_s + 4d^3 \mu \tau^2 - 4 \mu^2 d^2 \tau \tau^2 - 8 \mu d \tau L \dot{d} - 2 \mu^2 \tau \tau \dot{d} K_s]/[2d(\mu - 2d)^2 \mu \tau].
\]

As it can be seen, \( R_1(x_0) \) is constant on lines \( a + b = \text{constant} \), as it can be seen from the expression of \( R_{i1} \) and \( R_{i2} \) and observed in Fig. 12.

**VI. OPTIMIZATION PROBLEM IN THE CASE OF NEGLIGIBLE SETUP TIMES BUT NON-NEGLECTIBLE SETUP COSTS**

In this section we investigate the interesting scenario of a machine that may switch production from one part type to another with a negligible time still requiring a non negligible setup cost. A real world example can be a machine that may change tool with negligible time but every time needs a new tool, that is trashed after use. We give in this section only some guidelines of this interesting problem, starting from the analysis of Section 5 and only considering for simplicity the steady state optimization problem for the infinite buffer capacity case. Since setup times are negligible, as observed in Section 3, it is possible in this case to keep buffers empty. However this would imply at steady state an infinite cost since an infinite number of setups would be performed in a finite time interval. Considering the limit cycle studied in Section 5, it is expected that the optimal trajectory at steady state for the infinite buffer capacity case is the limit cycle represented in Fig. 13.

The limit cycle is as follows. From point \( A = (0, y_A) \) the system, with setup 1, meets the demand of part type 1 at rate \( d \) (it can not produce at a faster rate, since \( x_1 = 0 \)) up to point \( B = (0, y_B) \). The parameter \( l = \frac{y_B - y_A}{d} \) characterizes the limit cycle and its optimal value will be analytically derived next. After point \( B \), the system instantaneously performs a setup incurring in a setup cost \( K_s \). With a full production rate \( u_2 = \mu_2 \), the trajectory reaches \( C = (x_C, 0) \equiv (y_A, 0) \). From this point, the trajectory is symmetrically equivalent to the part of the limit cycle just described. The notation \( C_\mu(l) \) will denote a limit cycle with the shape reported in Fig. 13, with \( y_B - y_A = d \cdot l \). We have a result similar to Lemma 1, whose proof is also obtained by direct computation.

**Lemma 3.** Assume \( L \) is infinite. Then, the problem of finding

\[
l_* = \arg \min_{l \in (1)/20} J_u(l)
\]

is solved by

\[
l_* = (\mu - 2d) \sqrt{\frac{2K_s}{cd \mu (\mu - d)}}. \tag{27}
\]

Observe that Lemma 3 agrees with Lemma 1. In particular, the expression of \( l_1 \) in (22) becomes (27) if \( \mu \rightarrow 0 \).

**Remark 1.** As a remarkable difference with respect to the non negligible setup time case, it is interesting to observe that in this case \( l_* \) is always positive for any \( K_s > 0 \). That is, as soon as \( K_s > 0 \) the optimal trajectory leaves the origin.
VII. CONCLUSIONS

In this paper a dynamic scheduling problem for a single machine manufacturing system has been considered. The cost function comprises backlog and surplus, but also includes a demand loss component if buffers have a finite capacity and a setup cost if not negligible. The problem of dynamic scheduling for single machine manufacturing systems has been extensively studied in the past under different setup scenarios, mainly for systems with infinite buffer capacity. This paper addresses a general framework and investigates similarities and differences between policies optimal if setup times and costs are or are not negligible, if buffer capacities are finite or infinite. Both a steady state and a transient optimization problem are considered and already known results are compared and extended to complete the analysis. In particular, the following remarkable results have been shown in the paper: under all possible setup situations, the finite bound on the capacity buffer always implies, if the demand loss cost is small enough, a kind of inversion in the optimal solution of the infinite capacity case; moreover, an optimal trajectory moving along a border of the state space will be losing customers until a point of that border is reached, whose position is independent of initial conditions.

REFERENCES


Francesco Martinelli was born in Rome, Italy, in 1969. He received the Laurea degree magna cum laude in Electronic Engineering in 1994 and the Ph.D. degree in Computer Science and Control Engineering in 1998 both from the University of Rome Tor Vergata, Italy. During his Ph.D. studies, he spent approximately 8 months at the Department of Manufacturing Engineering at Boston University, MA, USA. In 1998 he obtained a scholarship at Fondazione U. Bordoni. Since October 1998 he is with the Department of Computer Science, Systems and Production of the University of Rome Tor Vergata as an Assistant Professor. His research interests include control of discrete event dynamic systems through the optimization of performance indexes, dynamic scheduling problems for manufacturing systems, motion planning of mobile robots, control theory regarding mechanical systems and non-linear systems.

Paolo Valigi received the Laurea degree (cum laude) in 1986 from the University of Rome La Sapienza, and the Ph.D. in 1990 from the University of Rome Tor Vergata. From 1991 to 1994 he has been researcher at Fondazione Ugo Bordoni, from 1994 to 1998 he has been associate researcher at University of Rome Tor Vergata, Department of Computer Science, Systems and Production, and since 1998 he is associate professor of System Theory at the Department of Electronic and Information Engineering, Faculty of Engineering, University of Perugia. He is member of IEEE, ANIPLA, CIRA (Consorzio Interuniversitario di Ricerche in Automatica) and COMES (COntrol and Models of Event Systems).