Finding Optimal Material Release Times Using Simulation-Based Optimization

Tito Homem-de-Mello • Alexander Shapiro • Mark L. Spearman
School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, Georgia 30332-0205

We present a method for setting release times for jobs with due dates in a stochastic production flow line for which the sequence of jobs has been determined. Unlike other approaches to this problem, ours considers a *transient* situation. Thus, the flow line will typically contain work in process (WIP), that is, jobs that have been previously released to the system.

Our goal is to develop a job release schedule that not only minimizes tardiness but also maximizes flexibility. The philosophy can be characterized as one that seeks to "release as late as possible, but no later!"

Our methodology is based on Monte Carlo simulation and consequent optimization by a method that became known as "stochastic counterpart" or "sample path" simulation-based optimization techniques. We use this method to minimize an expected value objective function that contains terms for tardiness and flow time "costs." We include a discussion of how the cost parameters of this objective function can be obtained by considering a "characteristic curve" for the system. We also discuss means for obtaining sensitivity analysis with respect to due dates and service times distributions parameters. We conclude with a numerical example.

(*Stochastic Systems*; *Simulation-Based Optimization*; *Flow Lines*; *Release Strategies*; *Tardiness*; *Lead Times*; *Manufacturing Executions Systems*; *Enterprise Resources Planning Systems*)

1. Introduction

Sales of Manufacturing Resources Planning (MRP II) and Enterprise Resource Planning (ERP) systems have climbed steadily in recent years. In 1989, MRP II sales accounted for almost a third of the total software market in the United States, with revenue of $1.2 billion (*IE* 1991). Last year, this total was exceeded by one company, SAP of Germany, with a total revenue of $1.8 billion ($367 million in the United States).

Unfortunately, at the heart of most of these systems is a scheduling module that relies on the same basic assumption of the original MRP systems designed almost 30 years ago—that of fixed planned lead times that depend only on the part being produced. Of course, as has been widely noted by scholars and practitioners alike, since capacity is finite, flow times depend on congestion. Because manufacturers typically desire high utilization of their resources, congestion can be high, and hence planned lead times must be long, leading to high inventory levels and sluggish customer responsiveness.

Recognition of this flaw in MRP II and ERP systems has triggered the recent flurry of development of Advanced Planning Systems (APS). These systems use finite capacity scheduling techniques that are based on a wide array of models that try to determine appropriate start times and schedules for jobs in recognition of capacity constraints. Growth in this area has been even more phenomenal than that in ERP, with the revenue of several APS vendors doubling every year for the past two or three years.

What has made all this advancement possible is the

0025-1909/99/4501/0086$05.00

development of what are called Manufacturing Execution Systems (MES). MES utilize emerging networking technology to provide real time tracking of status of manufacturing resources such as machines, labor, and tooling along with work-in-process (WIP). Ideally, an APS can download the current status of the shop floor from the MES, get planned order releases from the ERP system, and attempt to optimize the schedule. Unfortunately, the integration has not been so seamless, and the way APS, MES, and ERP systems will be coordinated is still being worked out (Gumaer 1996).

The addition of MES and APS to MRP II to make a truly comprehensive ERP systems has been a great improvement over older incarnations of MRP. However, no ERP system to date explicitly considers stochastic issues. Many allow for buffers either in the form of inventory or lead time, but they do not offer any suggestion as to how these buffers should be set.

This paper makes a step toward bridging this gap. While we cannot, at this point, consider extremely complex operations, we do examine a stochastic production line having existing WIP and for which the sequence but not the schedule of release of jobs is known. An example of such a situation would be a flow line in which the jobs all have roughly the same process time but have different characteristics and due dates (e.g., a flow line making circuit boards).

The model explicitly considers stochastic process times as well as down times. Conceivably the model could be used as an interface between the creation of the MRP pool and the MES job release module to determine exactly when jobs should be released. This would eliminate the need for a human planner to make such decisions. Criteria for release include both WIP levels and customer service (i.e., job tardiness). Parameters in the model can be adjusted to trade off between these competing desires. Finally, the model provides sensitivity analysis with respect to capacity and due dates. This gives a planner information regarding the cause of infeasibility of a given schedule.

2. Previous Work

Much of the work that has been done in this area has been related to material requirements planning (MRP) systems (see Vollman et al. 1988 for a comprehensive treatment of this subject). Under MRP the release times are given by subtracting a *planning lead time* from the due date of the job. This planning lead time is a constant that is stored in the MRP database and depends only on the part number.

The use of fixed lead times has led to large inventories. Why? Because, as one production control manager put it, "Customers can scream but inventory can't." Longer planning lead times help to ensure that jobs get finished in time, but they do so at the cost of more inventory and sluggish customer responsiveness. Also, since the lead times are often longer than what the customer will tolerate, forecasts must be used. This can lead to even more inventory (forecast high) and/or shortages (forecast too low).

To prevent excessive lead times, most MRP II systems provide two capacity checks: rough cut capacity planning (RCCP) and capacity requirements planning (CRP) along with an execution model known as Input/Output Control. RCCP provides a very rough check of the master production schedule by comparing demand against an aggregated load. CRP takes the planned order releases from MRP and projects their arrival throughout the plant. By adding these for all the releases, one can create a load profile at each process. However, CRP contains the same error as MRP, but it does it more often. CRP assumes a constant lead time at each process center rather than for the entire routing. For a complete discussion of the problems of MRP, RCCP, and CRP, see Hopp and Spearman (1996, ch. 3 and 5).

Input/output control simply monitors WIP and releases and alerts the planner whenever the WIP levels have increased beyond a certain level. However, this remedy is usually "too little, too late." The poor performance of MRP II systems was a large motivation for the rise in the use of so called Just-in-Time (JIT) systems during the 1980s. Instead of controlling releases via a master production schedule, JIT methods control WIP directly by either limiting it at each station (e.g., kanban) or by limiting it on a routing (e.g., CONWIP). It can be shown that controlling releases and measuring WIP (à la MRP II) is always less robust than controlling WIP and measuring output (à la kanban, CONWIP)—see Spearman and Zazanis (1992). Of course, while both kanban and CONWIP do help to smooth releases, neither is explicitly linked to due dates. Thus, without intervention, either system can sometimes pull jobs in too early and other times release jobs too late.

Nevertheless, there has been a fair amount of research in I/O Control and in setting lead times. Karni (1982) studies the basic equations of a deterministic I/O Control system and offers some insight into setting MRP lead times. Graves (1986) describes a stochastic model that allows flexible production rates in order to keep WIP levels (and, equivalently, lead times) small. Karmarkar (1987) describes a stochastic model of a single station that considers setup times and batch sizes in order to compute expected cycle times. The application of this model in an actual cell is described in Karmarkar et al. (1985).

Good surveys are provided by Baker, who discusses requirements planning, and Karmarkar, who describes issues surrounding manufacturing lead times (Graves et al. 1993, separate chapters).

Most of the papers dealing with setting release times use *steady state* models. There is much less literature dealing with transient systems. An exception is Saboo et al. (1989), who model the transient flow of materials in a generalized flow line using a set of recursive equations that involve the maximum of the arrival time of a job and the finish time of the job before it (similar to our model). Their performance measures are the expected make-span, delay in queues, station utilization, and lot tardiness. They approximate the maximum of two random variables with a bivariate normal and, consequently, require that the processing times at the stations be normally distributed. They state that approximation errors grow with an increase in the number of jobs and/or an increase in the number of stations. They also state that their algorithm substantially underestimates the variance of the finish times in most cases and conjecture that it is because they have ignored covariance effects.

Our model is similar to that used by Saboo et al. (1989). However, we will not attempt to approximate the maximum of two random variables. Instead, we will employ a rapid simulation methodology to compute individual sample paths.

3. Description of the Model

We consider a single stage in a production system in which there are K single-server stations and a set of J jobs that must be processed sequentially by all stations in a prescribed order. We assume that the processing of job j on station k is a random variable whose distribution is known, and that each station processes its coming jobs on a first-come-first-serve basis, holding waiting jobs in a queue of infinite capacity. Each job has a predetermined *due date* that should be met. We are also interested in keeping the flow time (or *cycle time*, as it is often referred to in industry and as we will call it in the remainder of the paper) of each job, which is given by the sum of the processing and the queueing time of that job over all K stations, as small as possible. Short cycle times are important because:

1. They provide more rapid response to customers.
2. They reduce work in process (WIP).
3. They result in less scrap because there typically is less time between defect creation and defect detection.

Also, in many cases shorter cycle times provide greater flexibility in manufacturing because raw materials (such as bar stock or blank wafers) do not receive "personality" before processing begins. Similarly, if the cycle times are short enough, an operation can reduce inventories by building to order rather than building to stock.

3.1. Example

The model described above would be appropriate for any manufacturing situation involving a flow line with jobs having due dates. One example is in a "raw card" circuit board plant. This operation begins by laminating one sheet of "pre-preg" between two sheets of copper to form a "core blank." The blank is then etched with circuitry to form a "core" in a sequence of operations called "core circuitize." Oftentimes, several cores are laminated together to form a "composite" board. Composite boards are then drilled and plated with copper to reconnect the various circuits on different layers.

Our model could be useful for releasing jobs in the core circuitize portion of the line. Jobs are released from an MRP system. Each job has, among other things, a part number denoting the size of board and the circuitry to be applied, a due date, and an order quantity. In one plant where one of the authors worked, there were over 5,000 varieties of circuit boards. Large customer orders are split into several jobs, each having the same part number and due date. Most of the jobs have the same number of boards limited by material handling devices (around 50 boards each). The core circuitize operation pulls core blanks from a supply (there are relatively few of these) along with a circuit master (there are many of these).

The first operation cleans the blank, followed by a step that applies a photo-resist material. This is followed by several mechanical operations that resize and punch the board. At this point the circuit is exposed onto the photo-resist material using ultraviolet light. This material is then developed in a chemical bath to fix the image. After development, the board is placed into another bath that etches off the copper where there was no image, after which another bath strips off any residual photo-resist material. The last operation is one to "inspect and repair" any "shorts" or "opens" that may have occurred.

Thus, the jobs are individual in that a different circuit is etched onto different boards. However, the process times are nearly identical for the jobs because they all have the same number of boards. Since the process times are nearly identical, the jobs are released in earliest due date order to maximize customer service.

One problem in this industry is that of long cycle times. Demand is composed of "firm orders" for specific customers, along with forecast demand. If cycle times are short enough, most of the demand will be in the form of firm orders. Obviously, less inventory can be carried in such a system. One way to reduce cycle times is to release jobs as late as possible. Before a blank is circuitized it can be used for up to 1000 different part numbers. However, once it has been exposed, it is committed to a specific part number. Hence the plant will keep a large stock of sheet copper and pre-preg but will attempt to keep WIP levels low. Thus the proposed model is perfectly suited for keeping cycle times short while meeting due dates.

3.2. Notation

We use the following notations and terminology throughout the paper. For a job j, ($j = 1, \ldots, J$) and a station k, ($k = 1, \ldots, K$) we denote by: S_{kj} the service time of processing job j on station k, C_{kj} the completion time, i.e., the time job j finishes its service at station k, d_j due date of job j, r_j the release time of job j, L_j^1 and L_j^2 the penalty cost for tardiness and the cost of holding job j in the factory per unit of time, respectively. We discuss how we obtain these costs below. By $S := (S_{11}, \ldots, S_{KJ})$ we denote the vector of service times, which is assumed to be random with a known distribution, and by $r := (r_1, \ldots, r_J)$ the vector of release times, which are viewed as the decision variables to be determined. Note that C_{Kj} can be viewed as a total completion time of job j in the queue and that each C_{kj} is a function of S, and of r, and hence is random.

The cost function, for processing J jobs, can be written then in the form

$$Z(r, S, d) := \sum_{j=1}^{J} \{L_j^1[C_{Kj} - d_j]^+ + L_j^2[(C_{Kj} \vee d_j) - r_j]\}, \tag{3.1}$$

where $a \vee b$ denotes the maximum of two numbers a and b, and $a^+ := a \vee 0$. Our objective is to minimize the expected value of the cost function subject to the order constraints $0 \leq r_1 \leq \cdots \leq r_J$. That is,

$$\min_{r \in \mathbb{R}^J} \{g(r, d) := \mathbb{E}\{Z(r, S, d)\}$$
$$\text{subject to } 0 \leq r_1 \leq \cdots \leq r_J. \tag{3.2}$$

Note that releasing jobs in a different order implies a different computation of completion times. Therefore the above constraints $r_1 \leq \cdots \leq r_J$ mean that the order in which jobs are released is *fixed*. Note also that the cost function, and its expected value, depend on the due dates $d := (d_1, \ldots, d_J)$. Sometimes we suppress d and write $Z(r, S)$, etc.

Let us observe at this point that since the max-operator preserves convexity, it follows that C_{Kj} is a convex function of r and S and hence $Z(r, S, d)$ is a piecewise linear *convex* function $Z : \mathbb{R}^J \times \mathbb{R}^{KJ} \times \mathbb{R}^J \to \mathbb{R}$. This implies that the expected value function $g(r, d) := \mathbb{E}\{Z(r, S, d)\}$ is convex for any distribution of S. Convexity is a very useful property in optimization. We discuss its implications for the present problem later.

3.3. Determination of Tardiness and Cycle Time Cost

One problem with the above formulation (and, indeed, many formulations in this literature) is the difficulty in determining the associated costs. Tardiness cost and cycle time cost are not intuitive. In order to understand this problem better, let us consider the following procedure.

Suppose that the penalty costs L_j^1 and L_j^2 do not depend on j, that is $L_j^1 = L^1$ and $L_j^2 = L^2$ for all j. Then all we need to specify is the ratio of these costs, call it $\lambda := L^1/L^2$. Then the objective becomes

$$Z(r, S, d) := \lambda \sum_{j=1}^{J} [C_{Kj} - d_j]^+ + \sum_{j=1}^{J} [(C_{Kj} \vee d_j) - r_j], \tag{3.3}$$

where again, we wish to minimize its expected value subject to the ordering constraints.

Let us observe that in this case the optimization problem (3.2) can be formulated in the following equivalent form:

$$\min_{r \in \mathbb{R}^J} \mathbb{E}\left\{ \sum_{j=1}^{J} [(C_{Kj} \vee d_j) - r_j] \right\}$$
$$\text{subject to } \mathbb{E}\left\{ \sum_{j=1}^{J} [C_{Kj} - d_j]^+ \right\} \le T,\ 0 \le r_1 \le \cdots \le r_J, \tag{3.4}$$

for some $T > 0$. Indeed

$$\mathcal{L}(r, \lambda) := \mathbb{E}\left\{ \sum_{j=1}^{J} [(C_{Kj} \vee d_j) - r_j] + \lambda\left(\sum_{j=1}^{J} [C_{Kj} - d_j]^+ - T \right) \right\}$$

is the Lagrangian of the above optimization problem. For fixed $\lambda > 0$, minimization of $\mathcal{L}(\cdot, \lambda)$, subject to $0 \le r_1 \le \cdots \le r_J$, is equivalent to solving optimization problem (3.2). Let r^* be an optimal solution of (3.2), i.e., r^* is a minimizer of $\mathcal{L}(\cdot, \lambda)$ for $\lambda := L^1/L^2$, and C_{Kj}^* be the corresponding completion times. Then r^* is also an optimal solution of (3.4) for

$$T := \mathbb{E}\left\{ \sum_{j=1}^{J} [C_{Kj}^* - d_j]^+ \right\}.$$

It also can be noted that, by convexity of (3.4) (see §3.4), the optimal value of the above problem (3.4) is a convex function of the right-hand-side perturbation parameter T.

From the above discussion we see that specifying costs for tardiness and cycle time (i.e., the ratio $\lambda := L^1/L^2$) is equivalent to dealing with the trade-off between those quantities. Suppose we plot a chart depicting the optimal expected tardiness against the optimal expected cycle time for different values of λ. By the above arguments, such chart will be the graph of a convex monotone function. The planner can then choose the values of expected tardiness and cycle time in the graph that are better suited to his priorities. We discuss this problem further in §8, where a numerical example is presented.

3.4. The Performance Model

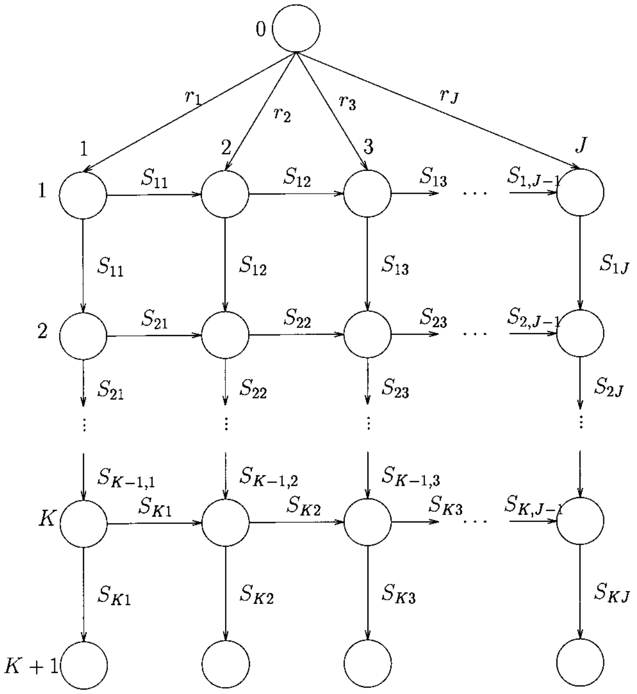
The completion times can be computed by the following recursive formula (see, e.g., Hasan and Spearman 1996):

$$C_{kj} = C_{k-1,j} \vee C_{k,j-1} + S_{kj},$$
$$j = 1, \ldots, J,\ k = 1, \ldots, K, \tag{3.5}$$

with $C_{k0} = 0$, $k = 1, \ldots, K$, and $C_{0j} = r_j$, $j = 1, \ldots, J$ (see Figure 1). It is worthwhile to note that the cost function can be also written in the following equivalent form:

$$Z(r, S, d) = \sum_{j=1}^{J} \{L_j^1(C_{Kj} \vee d_j - d_j) + L_j^2(C_{Kj} \vee d_j - r_j)\}$$
$$= \sum_{j=1}^{J} \{(L_j^1 + L_j^2)(C_{Kj} \vee d_j) - L_j^2 r_j - L_j^1 d_j\}. \tag{3.6}$$

Figure 1 **Graph Representation of Completion Times**



It will be convenient to represent processing (flow) of the jobs on the directed graph G given in Figure 1. We view the top node, labeled 0, as the beginning of the process, and the remaining nodes are labeled according to their position (k, j). The service times S_{kj} are viewed as distances between the corresponding nodes. The length of a path from the node 0 to a node (k, j) is given by the corresponding distance. For example, we can reach the node (2, 2) by traveling along the path $0 \to (1, 1) \to (1, 2) \to (2, 2)$. The length of that path is $r_1 + S_{11} + S_{12}$. The following proposition shows an equivalence between the longest paths and completion times.

Proposition 3.1. *Let P_{kj} be the length of the longest path from the node* 0 *to the node (k, j). Then, the following relation holds*:

$$C_{kj} = P_{kj} + S_{kj}. \tag{3.7}$$

Proof. We proceed by induction in k. Let us prove initially that (3.7) holds for $k = 1$, i.e., for the first row. For $j = 1$, clearly we have $P_{11} = r_1$ and hence (3.7) holds. Now suppose (3.7) holds for $j \geq 1$. The longest path from node 0 to node $(1, j + 1)$ is given by $P_{1,j+1} = (P_{1j} + S_{1j}) \vee r_{j+1}$. By the hypothesis of induction on j and (3.5), we have then $P_{1,j+1} = C_{1j} \vee r_{j+1} = C_{1,j+1} - S_{1,j+1}$, and hence (3.7) holds for $j + 1$.

Now suppose that (3.7) holds for rows $1, \ldots, k$, for some $k \geq 1$. We want to show that this relation also holds for row $k + 1$. Indeed, consider first node $(k + 1, 1)$. There is only one path from node 0 to that node, so we have $P_{k+1,1} = P_{k1} + S_{k1}$. From the induction hypothesis and (3.5) it follows that $P_{k+1,1} = C_{k1} = C_{k+1,1} - S_{k+1,1}$, hence (3.7) holds for $(k + 1, 1)$. Next, suppose for induction that (3.7) holds for nodes $(k + 1, 1), \ldots, (k + 1, j)$ for some $j \geq 1$, and consider node $(k + 1, j + 1)$. We have that

$$P_{k+1,j+1} = (P_{k,j+1} + S_{k,j+1}) \vee (P_{k+1,j} + S_{k+1,j}),$$

so by the hypotheses of induction (on the rows and on j) we have $P_{k+1,j+1} = C_{k,j+1} \vee C_{k+1,j}$ and hence it follows from (3.5) that $P_{k+1,j+1} = C_{k+1,j+1} - S_{k+1,j+1}$, so (3.7) holds for node $(k + 1, j + 1)$, thus completing the proof. □

In particular, it follows from (3.7) that the total completion time of job j can be written in the form

$$C_{Kj} = P_{K+1,j}. \tag{3.8}$$

Note also that the modified completion time $C_{Kj} \vee d_j$ can be easily computed through a longest path as well, just by adding an arc connecting node 0 to node $(K + 1, j)$ with the corresponding distance d_j. We shall use henceforth the term "total completion time" referring to $C_{Kj} \vee d_j$ and C_{Kj} interchangeably, the meaning being understood from the context.

Besides allowing a graphical representation of the completion times, the graph structure makes possible the use of efficient and well-known algorithms. In the present case the graph is *acyclic*, so we can use a *topological ordering* type of algorithm to find the longest path (see, e.g., Ahuja et al. 1993). The idea is to compute the longest path from the root node to all other nodes in the graph following the intrinsic topological order, and visiting each edge only once. Besides being simple and efficient, this kind of algorithm allows all completion times to be computed *simultaneously*. Note also that longest path problems can be written as linear programming problems (Ahuja et al. 1993). We describe that next.

We can think of the longest-path between nodes 0 and $(K + 1, j)$, as a maximum-cost flow between those nodes, where each arc has capacity equal to one. Consider the graph G depicted in Figure 1, augmented with the arc $(0, (K + 1, j))$. We use the following convention for the flow variables:

- v_{0j} denotes the flow on the arc connecting node 0 to node $(1, j)$, $j = 1, \ldots, J$;
- v_{kj} denotes the flow on the arc connecting node (k, j) to node $(k + 1, j)$, $k = 1, \ldots, K$, $j = 1, \ldots, J$;
- h_{kj} denotes the flow on the arc connecting node (k, j) to node $(k, j + 1)$, $k = 1, \ldots, K$, $j = 1, \ldots, J - 1$;
- u_j denotes the flow on the arc connecting node 0 to node $(K + 1, j)$, $j = 1, \ldots, J$.

By $v_0 := (v_{01}, \ldots, v_{0J})$, $v := (v_{11}, \ldots, v_{KJ})$, $h := (h_{11}, \ldots, h_{K,J-1})$ and $u := (u_1, \ldots, u_J)$ we denote the corresponding vectors.

It follows then that for every $l \in \{1, \ldots, J\}$ the total completion time $C_{Kl} \vee d_l$ is given by the optimal value of the following linear programming problem:

$$\max_{v_0, v, h, u_l} \left\{ \sum_{j=1}^{J} r_j v_{0j} + \sum_{j=1}^{J} \sum_{k=1}^{K} S_{kj}(v_{kj} + h_{kj}) + d_l u_l \right\}, \quad (3.9)$$

subject to

$$v_{k-1,j} + h_{k,j-1} = v_{kj} + h_{kj},$$
$$k = 1, \ldots, K, j = 1, \ldots, J, \quad (3.10)$$

$$u_l + \sum_{j=1}^{J} v_{0j} = 1, \quad (3.11)$$

$$u_l + v_{Kl} = 1, \quad (3.12)$$

$$v_{Kj} = 0, \quad j = 1, \ldots, l - 1, l + 1, \ldots, J, \quad (3.13)$$

$$0 \le v_{kj} \le 1, \quad k = 0, \ldots, K, j = 1, \ldots, J, \quad (3.14)$$

$$0 \le h_{kj} \le 1,$$
$$k = 1, \ldots, K, j = 1, \ldots, J - 1, \quad (3.15)$$

$$0 \le u_l \le 1. \quad (3.16)$$

A few words must be said about the above LP. Equation (3.10) reflects the conservation of the flow on nodes (k, j), $k = 1, \ldots, K$, $j = 1, \ldots, J$. Note that, by definition, $h_{k0} = 0$ and $h_{kJ} = 0$, $k = 1, \ldots, K$. Equation (3.11) defines node 0 as the *source* of the flow, whereas (3.12) and (3.13) define $(K + 1, l)$ to be the *sink* node. Finally, inequalities (3.14)–(3.16) are the capacity constraints on all arcs. Note that those inequalities can be replaced by $v_{kj}, h_{kj}, u_l \in \{0, 1\}$, since there exists an optimal solution of such problem that is integral (Ahuja et al. 1993).

3.5. Incorporating Failures into the Model

One source of randomness that often occurs in the considered type of problems results from machine *failures*. Breakdowns happen at random times, and the necessary repair also takes uncertain time due to the nature of the problem, availability of repairmen, etc. In this section we discuss how to incorporate this source of randomness into the model described in the previous sections. For the sake of simplicity, we assume that a broken machine starts to be repaired immediately after it fails, that repair times are independent from service times and failure epochs, and that chances that a machine fails more than one time, during service of a particular job, is negligible.

The basic idea is to "inflate" the service times by mixing their distributions with the repair times distributions, using the probability of failure as a weight. Formally, let S_{kj} be the service time of job j at station k, and denote by R_k the repair time of station k. Let $R_{k1}, \ldots, R_{kJ}$ be iid random variables with the same distribution as R_k. Finally, let p_{kj} be the probability that station k fails given that job j is being served there, and assume that this probability depends only on k and j. The "inflated" service time $\tilde{S}_{kj}$ is defined as

$$\tilde{S}_{kj} := S_{kj} + Y_{kj} R_{kj}, \quad (3.17)$$

where Y_{kj} is a Bernoulli random variable taking value 1 with probability p_{kj} and 0 with probability $1 - p_{kj}$. Such inflated times $\tilde{S}_{kj}$ can be easily generated. To do so, we generate S_{kj}, Y_{kj} and R_{kj} *independently* according to the respective distributions, and then compute $\tilde{S}_{kj}$ as in Equation (3.17). Now, $\tilde{S}_{kj}$ is taken to be the "new" service time, that is, $\tilde{S}_{kj}$ replaces S_{kj} in the model described above.

The probabilities p_{kj} can be computed (or at least estimated) if the *times between failures* of each station are independent and exponentially distributed. Indeed, let F_k be a random variable representing the time between successive failures at station k. By the memoryless property of the exponential distribution, the distribution of the time until next failure from the moment job j starts its service is still exponentially distributed with the same parameter, so the probability that station k fails before completing the service for job j is given by $p_{kj} = P(F_k \le S_{kj})$. This probability can sometimes be computed analytically, otherwise it can be estimated by simulation.

4. Differentiability Properties of the Expected Value Function

In this section we discuss differentiability properties of the expected value function $g(r) := \mathbb{E}\{Z(r, S)\}$ (we drop the parameter vector d in order to ease the notation). Calculation (estimation) of the first, and possibly second, order derivatives of $g(r)$ is a starting point of any efficient approach to the optimization problem (3.2). Since the function $g(r)$ is convex, we can use powerful tools of convex analysis. Recall that a vector $v \in \mathbb{R}^J$ is said to be a *subgradient* of $g(\cdot)$, at a point r, if for all $r' \in \mathbb{R}^J$:

$$g(r') - g(r) \ge v^T(r' - r).$$

The set of all subgradients of g at r is called the *subdifferential* of g at r and denoted $\partial g(r)$. The real-valued convex function g is differentiable at r if and only if the set $\partial g(r)$ is a singleton, i.e., contains only one element. In the latter case this element (subgradient) coincides with the gradient of g at r (see Rockafellar 1970, Theorem 25.1, for details). Note that a real valued convex function is locally Lipschitz continuous, which implies that the concepts of Gâteaux and Fréchet differentiability are equivalent for such functions.

It is possible to show that

$$\partial g(r) = \mathbb{E}\{\partial Z(r, S)\}, \tag{4.1}$$

where the subdifferential inside the expected value in (4.1) is taken with respect to r. The expected value $\mathbb{E}\{F(S)\}$ of the set-valued mapping $F(S) := \partial Z(r, S)$ is understood as the set of points given by expectations of measurable and integrable selections of F (see Ioffe and Tihomirov 1979, §8.3; Rockafellar 1968, Rockafellar and Wets 1982 for details and proofs). The required regularity conditions for (4.1) to hold are very mild. Apart from some measurability assumptions (which certainly hold in the present case) it is required only that the expected value function $g(r) = \mathbb{E}\{Z(r, S)\}$ be finite valued.

An important consequence of (4.1) is that $\partial g(r)$ is a singleton if and only if $\partial Z(r, S)$ is a singleton for almost every S (with respect to the probability measure of S). That is, g is differentiable at r if and only if $Z(\,\cdot\,, S)$ is differentiable at r with probability one. In the last case

$$\nabla g(r) = \mathbb{E}\{\nabla Z(r, S)\}, \tag{4.2}$$

where the gradient $\nabla Z(r, S)$ is taken with respect to r. We show now that if the distribution of the random vector S (of service times) has a density function, then indeed $Z(\,\cdot\,, S)$ is differentiable with probability one, and hence (4.2) holds at every point r.

The following result, due to Danskin (1967), will be useful in several respects. Consider a real valued function $\varphi(x, y)$, $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, a set $C \subset \mathbb{R}^m$ and the corresponding max-function $\psi(x) := \sup_{y \in C} \varphi(x, y)$.

THEOREM 4.1. *Suppose that for all $y \in C$ the function $\varphi(\,\cdot\,, y)$ is differentiable, that $\varphi(x, y)$ and $\nabla_x \varphi(x, y)$ are continuous on $\mathbb{R}^n \times C$ and that the set C is compact. Then the max-function $\psi(x) := \sup_{y \in C} \varphi(x, y)$ is directionally differentiable and its directional derivatives $\psi'(x, d)$ are given by*

$$\psi'(x, d) = \max_{y \in C^*(x)} d^T \nabla_x \varphi(x, y), \tag{4.3}$$

where $C^*(x) := \arg\max_{y \in C} \varphi(x, y)$ is the set of maximizers of $\varphi(x, \cdot\,)$ over C.

In particular the above theorem implies that if the set $C^*(x) = \{\bar{y}\}$ is a singleton, i.e., $\varphi(x, \cdot\,)$ has the unique maximizer $\bar{y}$ over C, then the max-function $\psi(\cdot)$ is differentiable at x and

$$\nabla \psi(x) = \nabla_x \varphi(x, \bar{y}). \tag{4.4}$$

Note that, under the assumptions of Theorem 4.1, the max-function $\psi(\cdot)$ is locally Lipschitz continuous, and hence the concepts of Gâteaux and Fréchet differentiability are equivalent.

Consider now the linear program (3.9)–(3.16). The optimal value of this LP gives the total completion time $\psi_l := C_{Kl} \vee d_l$ and can be viewed as the optimal value function

$$\psi_l(r, S, d) := \max_{(v_0,w,u)\in C} \varphi(r, S, d, v_0, w, u), \quad (4.5)$$

where

$$\varphi(r, S, d, v_0, w, u) := r^T v_0 + S^T w + d^T u,$$

$w := v + h$, and C is the corresponding feasible set. This feasible set C is defined by linear constraints, obtained from (3.10)–(3.16) by the transformation $v + h \rightarrow w$, and hence is a convex polygon. Moreover, it is not difficult to see that the set C is nonempty and, because of the constraints (3.14)–(3.16), is bounded and hence is compact. It follows then by Theorem 4.1 that the function ψ_l, defined in (4.5), is differentiable at a point (r, S, d) if (and only if) the LP (3.9)–(3.16) has a unique optimal solution $(\bar{v}_0, \bar{w} := \bar{v} + \bar{h}, \bar{u})$ (that is, if $(\bar{v}_0, \bar{v}, \bar{h}, \bar{u})$ is an optimal solution of (3.9)–(3.16), then $\bar{v}_0$, $\bar{w} = \bar{v} + \bar{h}$ and $\bar{u}$ are unique). In the last case

$$\nabla_r \psi_l(r, S, d) = \bar{v}_0, \quad (4.6)$$

and similarly for the gradients with respect to S and d.

Let us observe now that since C is a convex polygon, the set of those vectors (r, S, d) for which the corresponding LP has *more* than one optimal solution is formed by the union of a finite number of affine subspaces of the space $\mathbb{R}^J \times \mathbb{R}^{KJ} \times \mathbb{R}^J$. In particular, for any fixed r and d, the set $\mathcal{F}(r, d)$ of vectors S for which the corresponding LP has more than one optimal solution, is a union of a finite number of affine subspaces of $\mathbb{R}^{KJ}$. Since a proper affine subspace of a finite dimensional vector space has Lebesgue measure zero, we obtain that the set $\mathcal{F}(r, d)$ has Lebesgue measure zero in $\mathbb{R}^{KJ}$. (An affine subspace is said to be proper if it does not coincide with the whole space.) It follows that if the random vector S of the service times has a probability density function (pdf), then for any r and d the cost function $Z(\cdot, S, \cdot)$ is differentiable at (r, d) for almost every S (with respect to the probability distribution of S). Consequently, in that case formula (4.2) holds, i.e., the operator of differentiation can be taken inside the expected value.

Let us summarize the above discussion in the following proposition.

Proposition 4.1. *Suppose that the random vector S, of service times, has a continuous distribution described by a probability density function. Let $\mathcal{P}_l$ be the set of arcs that form the longest path from node 0 to node $(K + 1, l)$ on the graph $G \cup$ arc$(0, (K + 1, l))$ with weights (distances) (r, S, d). Then:* (i) *for any given r and d, the total completion times $C_{Kl} \vee d_l$, $l = 1, \ldots, J$, are differentiable at (r, d) with probability one, and*

$$\frac{\partial C_{Kl} \vee d_l}{\partial r_j}(r, S, d) = \begin{cases} 1, & \text{if arc } (0, (1, j)) \in \mathcal{P}_l, \\ 0, & \text{otherwise,} \end{cases} \quad (4.7)$$

$$\frac{\partial C_{Kl} \vee d_l}{\partial d_j}(r, S, d) = \begin{cases} 1, & \text{if } j = l \text{ and arc } (0, (K + 1, l)) \in \mathcal{P}_l, \\ 0, & \text{otherwise.} \end{cases} \quad (4.8)$$

(ii) *The expected values of these completion times, and hence the expected value of the cost function, are differentiable functions of r and d and the above partial derivatives can be taken inside the expected value operator.*

In particular, if the service times S_{kj} are mutually independent and each has a density function, then the corresponding random vector S has a continuous distribution with the corresponding pdf given by the product of the pdfs of the S_{kj}.

Now let $S^1, \ldots, S^N$ be a generated random sample of N independent realizations of the service times vector S. Then we can estimate the expected value function $g(r, d) := \mathbb{E}\{Z(r, S, d)\}$ by the sample average function

$$\hat{g}_N(r, d) := N^{-1} \sum_{i=1}^{N} Z(r, S^i, d). \quad (4.9)$$

By the Law of Large Numbers, for any given r and d, $\hat{g}_N(r, d)$ converges with probability 1 to $g(r, d)$ as $N \rightarrow \infty$. That is, $\hat{g}_N(r, d)$ is a consistent estimator of $g(r, d)$. Moreover, we have by Proposition 4.1(ii) that $\nabla g(r, d) = \mathbb{E}\{\nabla Z(r, S, d)\}$, provided S has a continuous distribution. By the Law of Large Numbers this implies that $\nabla \hat{g}_N(r, d)$ converges to $\nabla g(r, d)$ with probability 1. Note the partial derivatives

$$\frac{\partial \hat{g}_N}{\partial r_l}(r, d) = N^{-1}\left\{ \sum_{i=1}^{N} \sum_{j=1}^{J} (L_j^1 + L_j^2) \cdot \frac{\partial C_{Kj} \vee d_j}{\partial r_l}(r, S^i, d) - L_l^2 \right\} \quad (4.10)$$

at those points where $\hat{g}_N$ is differentiable, and similarly for the partial derivatives with respect to d_l.

5. Lower Bounds

Consider the expected values (means) $\mu_{kj} := \mathbb{E}\{S_{kj}\}$ and the corresponding vector $\mu = (\mu_{11}, \ldots, \mu_{KJ})$. By Jensen's inequality it follows from convexity of $Z(r, \cdot, d)$ that for any r and d,

$$\mathbb{E}\{Z(r, S, d)\} \geq Z(r, \mu, d). \quad (5.1)$$

This implies that the optimal value of the problem (3.2) is greater than or equal to the optimal value of

$$\min_{r \in \mathbb{R}^J} Z(r, \mu, d) \quad \text{subject to } 0 \leq r_1 \leq \cdots \leq r_J. \quad (5.2)$$

Let us show next that the optimization problem (5.2) can be formulated as an LP problem.

Consider the graph in Figure 1 with weights r and $\mu = S$, and let R_{ij} be the length of the longest path from node $(1, i)$ to node $(K + 1, j)$. Define $R_{ij} = -\infty$ whenever such path does not exist, i.e. $i > j$. Clearly, the length of the longest path from node 0 to node $(K + 1, j)$ is given by $\max_{1 \leq i \leq j}\{r_i + R_{ij}\}$, and hence by Proposition 3.1 we have that

$$C_{Kj} \vee d_j = (r_1 + R_{1j}) \vee \cdots \vee (r_j + R_{jj}) \vee d_j. \quad (5.3)$$

Notice that the matrix $R := [R_{ij}]$ is *constant* in the sense that it does not depend on r and d, although it depends on μ. Now the right-hand side of (5.3) can be written as the LP problem

$$\min z_j \quad \text{s.t. } z_j \geq r_i + R_{ij},\ i = 1, \ldots, j,\ z_j \geq d_j. \quad (5.4)$$

Consequently, by using nonnegativity of the costs L_j^1, L_j^2, we obtain that problem (5.2) can be formulated in the form

$$\min_{z,r}\left\{ \sum_{j=1}^{J} (L_j^1 + L_j^2) z_j - \sum_{j=1}^{J} L_j^2 r_j - \sum_{j=1}^{J} L_j^1 d_j \right\}$$

$$\text{subject to } z_j - r_i \geq R_{ij},\ i = 1, \ldots, j,\ j = 1, \ldots, J,$$
$$z_j - d_j \geq 0,\ j = 1, \ldots, J,$$
$$r_{j+1} - r_j \geq 0,\ j = 1, \ldots, J - 1,$$
$$r_1 \geq 0. \quad (5.5)$$

The above LP problem (5.5) can be solved relatively easily. Its optimal value gives a lower bound for the optimal value of the original problem (3.2) and corresponds to the situation where the service times are replaced by their means.

Another lower bound can be constructed as follows. Let $S^1, \ldots, S^N$ be a generated random sample of the service times vector S. For each S^i consider the associated problem

$$\min_{r \in \mathbb{R}^J} Z(r, S^i, d) \quad \text{subject to } 0 \leq r_1 \leq \cdots \leq r_J. \quad (5.6)$$

By the above arguments problem (5.6) can be formulated as an LP problem in a way similar to (5.5) (by replacing the weights μ with S^i), and hence its optimal value Z_i^* can be easily calculated. By the Law of Large Numbers we have then that $\hat{Z}_N := N^{-1} \sum_{i=1}^{N} Z_i^*$ converges with probability one to $\mathbb{E}\{\min_{r \in \mathcal{R}} Z(r, S, d)\}$, where $\mathcal{R} := \{r \in \mathbb{R}^J : 0 \leq r_1 \leq \cdots \leq r_J\}$. Let us observe now that for any $r \in \mathcal{R}$,

$$\mathbb{E}\{Z(r, S, d)\} \geq \mathbb{E}\{\min_{r \in \mathcal{R}} Z(r, S, d)\},$$

and hence

$$\min_{r \in \mathcal{R}} \mathbb{E}\{Z(r, S, d)\} \geq \mathbb{E}\{\min_{r \in \mathcal{R}} Z(r, S, d)\}. \quad (5.7)$$

We have here that the right-hand side of (5.7) gives a lower bound for the optimal value of problem (3.2) and that $\hat{Z}_N$ is a consistent estimator of that lower bound.

We also have the following inequality relation between the above two bounds:

$$\mathbb{E}\{\min_{r \in \mathcal{R}} Z(r, S, d)\} \geq \min_{r \in \mathcal{R}} Z(r, \mu, d). \quad (5.8)$$

Indeed, since the function $Z(\cdot, \cdot, \cdot)$ and the set $\mathcal{R}$ are convex, it follows that the function $\zeta(S, d) := \min_{r \in \mathcal{R}} Z(r, S, d)$ is also convex. Inequality (5.8) is then a consequence of Jensen's inequality.

6. Numerical Methods

There are basically two approaches to a numerical solution of the problem (3.2) by using Monte Carlo estimators (4.9) and (4.10) of the value $g(r)$ and the gradient $\nabla g(r)$, respectively, of the expected value function at a point r. One approach is based on the stochastic approximation (SA) method (see, e.g., Kushner and Clark 1978). The SA method generates iterates by the following procedure:

$$r^{k+1} = \Pi_{\mathcal{R}}(r^k - a_k \gamma_k), \tag{6.1}$$

where γ_k is an estimate of the gradient $\nabla g(r^k)$, a_k is a chosen sequence of positive numbers and $\Pi_{\mathcal{R}}$ denotes the projection operator onto the feasible set $\mathcal{R}$. For example one can use the estimate $\gamma_k := \nabla \hat{g}_N(r^k)$, given in (4.10), for a generated sample. The numbers (stepsizes) a_k typically are chosen a priori, and one has little control over a_k in the process of optimization.

In this paper we discuss a variant of an alternative approach, which became known as a stochastic counterpart or sample path method (cf. Rubinstein and Shapiro 1993; Plambeck et al. 1993, 1996). The basic idea of the method is simple indeed. A large sample is generated and then the corresponding sample average function $\hat{g}_N(r)$, given in (4.9), is minimized by deterministic methods of nonlinear programming. Of course, an implementation of that idea requires specification of a particular algorithm which is used for minimization of $\hat{g}_N(r)$. Typically calculations of the value $\hat{g}_N(r^k)$ and its gradient $\nabla \hat{g}_N(r^k)$ of the sample average function, at a current iteration point r^k, are time consuming and their computational time is proportional to the size N of the generated sample. In the beginning of the process, when the iterates are far away from the optimum, it does not make sense to generate a large sample since there even a relatively small sample allows to make a significant progress. Eventually the sample size should be increased in the process of optimization in order to obtain a better accuracy of the final estimate of the optimal solution (cf. Shapiro and Homem-de-Mello 1998).

We now proceed to the description of the algorithm. In order to ease the understanding, we outline it in blocks, which will be described later in detail.

1. Choose a (small) sample size $N = N_1$ and initial value r^1 of the vector of release times.
2. For a current iterate k, generate a sample, of size $N = N_k$, of service times vectors $S^1, \ldots, S^N$, and compute the estimators $\hat{g}_N(r^k)$ and $\nabla \hat{g}_N(r^k)$ according to (4.9) and (4.10), respectively.
3. Compute a descent direction δ_k by projecting the estimator $\nabla \hat{g}_N(r^k)$ onto an appropriate space and compute a stepsize α_k by a (crude) line search.
4. Set $r^{k+1} := r^k + \alpha_k \delta_k$ and compute $\hat{g}_N(r^{k+1})$ and $\nabla \hat{g}_N(r^{k+1})$ using the same sample $S^1, \ldots, S^N$.
5. If the decrease $\hat{g}_N(r^k) - \hat{g}_N(r^{k+1})$ is significantly large, then go back to Step 3 with $k \to k + 1$ and $r^k \to r^{k+1}$. Otherwise, generate a new sample $\tilde{S}^1, \ldots, \tilde{S}^N$ and compute new estimators $\tilde{g}_N(r^{k+1})$, $\nabla \tilde{g}_N(r^{k+1})$.
6. Compute new appropriate sample size N_{k+1} and extend sample to $\tilde{S}^1, \ldots, \tilde{S}^{N_k}, \tilde{S}^{N_k+1}, \ldots, \tilde{S}^{N_{k+1}}$. Compute $\tilde{g}_{N_{k+1}}(r^{k+1})$, $\nabla \tilde{g}_{N_{k+1}}(r^{k+1})$.
7. Test statistical stopping criteria based on the difference $\hat{g}_{N_k}(r^k) - \tilde{g}_{N_{k+1}}(r^{k+1})$ and on the estimator $\nabla \tilde{g}_{N_{k+1}}(r^{k+1})$. If none of them is satisfied, go to Step 3 with $k \to k + 1$, $r^k \to r^{k+1}$ and $N_k \to N_{k+1}$. Otherwise, STOP.

Following the above outline, we now proceed to the detailed description of each step.

STEP 1. The initial sample size should be small enough in order to accelerate the first iterations. In our implementation we take $N = 50$. The initial release times can be set arbitrarily. A "good" initial guess can be obtained by setting the release times as the optimal solution of the problem (5.2). Recall that the optimal value of (5.2) provides a lower bound for the optimal value of the problem (3.2).

STEP 2. For each S^i, solve the longest-path problem for the graph with weights (r^k, S^i, d) using a topological ordering algorithm (see the discussion in §3.4). Recall that all terms C_{Kj} as well as the corresponding derivatives can be computed simultaneously.

STEP 3. The stepsize can be computed by using any first-order optimization algorithm. In our case, we project the estimator $\nabla \hat{g}_N(r^k)$ onto the null space of the matrix corresponding to the constraints $\{r_j \le r_{j+1}, r_1 \ge 0\}$ that are active at r^k, and then apply a *line search* to find the stepsize, using Armijo's algorithm. See, e.g., Bazaraa et al. (1993) for a detailed description of this method. The choice for a projection algorithm was driven mainly by the simplicity of the constraints and by the fact that in many cases the constraints are never active during the iterations of the algorithm.

STEP 4. The computation is analogous to Step 2, but now the corresponding graph i has weights (r^{k+1}, S^i, d). Observe that one could compute the completion times by constructing for each S^i a matrix R^i similar to the one used in Step 1 (i.e., depending only on the weights S^i). In this case, the computation of the completion times for r^{k+1} would be extremely fast, since the matrices R^i would not have to be computed again. The price, of course, is the computation (J^2 longest-paths) and storage of all matrices R^i. This seems to be impractical in situations where the number of jobs is large, so we adopt the same procedure as in Step 2.

STEP 5. The significance of the decrease in the value of the function $\hat{g}_N(\cdot)$ is measured by a statistical paired t-test (see Shapiro and Homem-de-Mello 1998 for details).

STEP 6. Two tests are performed in order to determine the new sample size. The first one calculates the sample variance $\hat{\sigma}_N^2$ of $\hat{g}_N(r^{k+1})$ and verifies the condition

$$z_{\alpha/2}\frac{\hat{\sigma}_N}{\sqrt{N}} < \beta\hat{g}_N(r^{k+1}) \tag{6.2}$$

for some prespecified β. If the above condition holds, the current sample size is kept. Otherwise, we want to choose the new sample size N' such that

$$z_{\alpha/2}\frac{\hat{\sigma}_N}{\sqrt{N'}} < \beta\hat{g}_N(r^{k+1}),$$

and hence we can take

$$N' = \left\lceil \left(\frac{z_{\alpha/2}\hat{\sigma}_N}{\beta\hat{g}_N(r^{k+1})}\right)^2 \right\rceil,$$

where $\lceil a \rceil$ denotes the integer part of a.

The second test aims to guarantee (up to some specified confidence) that the estimator $\nabla\hat{g}_N(r^{k+1})$ will indeed yield a descent direction for the original problem. To do so, a confidence region

$$R = \{z \in \mathbb{R}^J : (z - \nabla\hat{g}_N(r^{k+1}))^T\hat{\Sigma}_N^{-1}(z - \nabla\hat{g}_N(r^{k+1})) \le \chi_J^2(\alpha)/N\}$$

(where $\hat{\Sigma}_N$ is the sample covariance matrix) is computed, and it is imposed that every vector in that confidence region make a nonnegative scalar product with $\nabla\hat{g}_N(r^{k+1})$. In either of the tests, the correction factor is limited to four times. We again refer to Shapiro and Homem-de-Mello (1998) for a more detailed description of the test.

STEP 7. The reduction in the value of the function is again verified by a statistical t-test (note that here the estimates $\hat{g}_{N_k}(r^k)$ and $\tilde{g}_{N_{k+1}}(r^{k+1})$ are independent).

The second stopping criterion measures the quality of the current solution r^{k+1} by computing the distance from the gradient estimator $\nabla\tilde{g}_{N_{k+1}}(r^{k+1})$ to the *optimal cone* corresponding to the set of points which satisfy the Karush-Kuhn-Tucker optimality conditions. The conditions for the applicability of such test as well as its detailed description can be found in Shapiro and Homem-de-Mello (1998). Let a_i denote the ith row of the matrix corresponding to the constraints $r_{j+1} - r_j \ge 0$ (i.e. $a_i = (0, \ldots, 0, -1, 1, 0, \ldots, 0)$), and let $I(r)$ denote the set of active constraints at r, that is,

$$I(r) = \{j : r_j = r_{j+1}, 1 \le j \le J - 1\}.$$

Let $N = N_{k+1}$, and consider the statistic

$$T := \min_{z\in C}(\nabla\tilde{g}_N(r^{k+1}) - z)^T\hat{\Sigma}_N^{-1}(\nabla\tilde{g}_N(r_{k+1}) - z),$$

where $\hat{\Sigma}_N$ is the sample estimator of the covariance matrix of $\nabla\tilde{g}_N(r^{k+1})$, and C is the cone

$$C := \left\{ z \in \mathbb{R}^J : z = \sum_{i\in I(r^{k+1})} \lambda_i a_i,\ \lambda_i \ge 0,\ i \in I(r^{k+1}) \right\}.$$

It is known that under mild regularity conditions, the null distribution of T (i.e., the distribution of T under the null hypothesis: "r^{k+1} is an optimal solution") is a central chi-square with $J - p$ degrees of freedom, where p is the number of active constraints at r^{k+1} (Shapiro and Homem-de-Mello 1998). Hence, given a prespecified significance level α, we compute c such that

$$\text{Prob}\{\chi^2_{J-p} \geq c\} = \alpha,$$

and hence accept the optimality of r^{k+1} if $T \leq c$.

7. Sensitivity Analysis

In optimization problems, one often faces the question of how much an obtained solution would change if some input data were slightly perturbed. In the present context of optimization of release times that situation may arise, for example, if the due date of some job is either postponed or anticipated because of a re-order from the client. It also could happen that one is interested in analyzing whether it is worthwhile improving the capacity of one or more stations in terms of reduction of tardiness and cycle-times of jobs. That is, what would be an effect of changing some parameter of the distribution of service times in the objective function previously computed? Such questions amount to studying the derivatives of the optimum value function and the optimal solutions with respect to the desired parameters, using the tools provided by the theory of *sensitivity analysis*.

The approach used in this work to solve the original optimization problem (3.2) also allows a computation of these derivatives under some conditions. Suppose initially that we want to compute the sensitivity of the optimal value of the problem (3.2) with respect to the due-dates d. Consider the associated optimal value function

$$\nu(d) := \min_{r \in \mathcal{R}} g(r, d). \tag{7.1}$$

Let us first observe that the function $\nu(d)$ is *convex*. This follows at once from convexity of $g(\,\cdot\,,\,\cdot\,)$ and convexity of $\mathcal{R}$. Second, suppose that the problem (3.2) has a unique optimal solution r^*. Since the problem (3.2) is convex, this holds if the function $g(\,\cdot\,, d)$ is strictly convex. Then by Theorem 4.1 we have that $\nu(\cdot)$ is differentiable and

$$\frac{\partial \nu(d)}{\partial d_j} = \frac{\partial g(r^*, d)}{\partial d_j}, \quad j = 1, \ldots, J. \tag{7.2}$$

The partial derivatives in the right-hand side of (7.2) can be estimated from the generated sample $S^1, \ldots, S^N$ by using formula (4.8). That is, if $\hat{r}$ is an estimate of r^* (obtained by solving (3.2)), then $\partial \hat{g}_N(\hat{r}, d)/\partial d_j$ gives an estimate of $\partial \nu(d)/\partial d_j$, $j = 1, \ldots, J$. In a similar way, one can estimate derivatives of the optimal value of (3.2) with respect to parameters involved in the distributions of the service times S_{kj}.

Note that the sensitivities with respect to due dates provide useful information about the probability of tardiness. Indeed, from (3.6), (4.8), and (4.9) it follows that

$$\frac{\partial \hat{g}_N(\hat{r}, d)}{\partial d_j} = N^{-1}\Bigg\{ \sum_{i=1}^{N} (L_j^1 + L_j^2) \cdot \frac{\partial C_{Kj} \vee d_j}{\partial d_j}(\hat{r}, S^i, d) - L_j^1 \Bigg\}$$

$$= N^{-1}\Bigg\{ \sum_{i=1}^{N} (L_j^1 + L_j^2) 1_{\{C_{Kj}(\hat{r}) \leq d_j\}} - L_j^1 \Bigg\},$$

and hence by the Strong Law of Large Numbers we have that

$$\frac{L_j^1 + \partial \nu(d)/\partial d_j}{L_j^1 + L_j^2}$$

$$= P(C_{Kj}(\hat{r}) \leq d_j)$$

$$= 1 - P(\text{job } j \text{ is tardy} \mid \text{release times} = \hat{r}). \tag{7.3}$$

The above computations allow to estimate the change in the optimum value of (3.2) when certain parameters are perturbed. A more complicated issue concerns the sensitivity of the *optimal solution* of that problem with respect to the same parameters. That is, one may ask how the optimal solution r^* varies as the input data changes. It is possible to show that if r^* is unique and is an interior point of $\mathcal{R}$, then

$$\nabla r^*(d) = -[\nabla^2_{dd} g(r^*, d)]^{-1}(\nabla^2_{dr} g(r^*, d)). \tag{7.4}$$

(see Fiacco 1983 for details). An application of the above formula requires calculation (estimation) of the second order derivatives of $g(r, d)$, which may be not easy. This problem requires a further investigation.

8. Numerical Results

In this section we present some numerical results obtained with an implementation of the algorithm

Table 1 Process Times and Mean Times to Failure and Repair for Example Problem

	Process Times (Min.)				
Station	Mean	Std Dev	Mean Time to Failure (Hours)	Mean Time to Repair (Hours)	Probability of Failure
1	20	1	5	2	0.0645
2	25	2.5	16	2	0.0257
3	24	1.5	2.5	1	0.1478
4	16	1.5	9	2	0.0292
5	20	4.5			

described in §6, applied to the example problem discussed in §3.1. Consider a line with 5 processing stations (each one corresponding to one operation of the core circuitize process) and 25 jobs, and suppose the planning is made at the beginning of the day. The first 10 jobs are due at the end of that day (8 hours), and the remaining 15 jobs are due at the end of the next day (16 hours). The data for the stations are given in Table 1. The distribution of service times is assumed to be normal, whereas the time to failure and the time of repair are assumed to be exponentially distributed. The last column displays the approximate probability of failure of each station.

These data were used with values for the ratio $\lambda := L^1/L^2$ (see §3.2) ranging from 0.1 to 1000. For each of six chosen values for λ (0.1, 0.5, 1, 10, 100, 1000) we ran the program to compute the optimal solution. The full line curve on Figure 2 shows the relation between the corresponding expected tardiness and expected cycle time per job. Observe the convexity of the graph, which was expected in light of the remarks made in §3.2. The chart can help the planner to decide which value of λ is appropriate for the case under study, based on the relative importance of tardiness and cycle time attributed by the user. For example, if the planner chose the point corresponding to an expected cycle time of around 300 and tardiness of around 125 as a reasonable trade-off level, the value of λ to use would be 1.0.

For the sake of comparison of the release policy suggested by our method with other policies, we considered the case when jobs are released according to fixed *lead-times* (see §1). In that policy, given a value of a parameter l, job j is released at time $[d_j - l]^+$, where d_j is the due date of that job. The dashed-and-dotted line in the graph shows the expected tardiness and expected cycle time computed for values of l equal to 120, 240, 480, 600, 720, and 960 (in minutes). That curve (of the lead-time policy) is also the graph of a convex function. In order to see that, consider the constrained problem (3.4) where the optimization now is performed with respect to the lead time l. For a given lead-time $\bar{l}$, let T be the corresponding expected tardiness. Then, since the expected cycle time is a monotonically increasing function of l and the expected tardiness is a monotonically decreasing function of l, we obtain that the constraint of the considered optimization problem is active at the optimal solution l^*, i.e., the expected tardiness is equal T for $l = l^*$. It follows that $\bar{l} = l^*$, i.e., for that choice of T, $\bar{l}$ is the optimal solution. Since the functions $[d_j - l]^+$ are linear in l on the interval $(0, \min_j\{d_j\})$, the convexity of the curve of the lead-time policy on that interval follows. With some additional effort it is possible to show that the above convexity holds on the larger interval $(0, \max_j\{d_j\})$ as well.

Clearly, the curve obtained with lead-times is always above the curve given by the optimal solution with arbitrary release times. In other words, given the release times corresponding to some value of l there exists a different set of release times that yields less

Figure 2 Expected Tardiness vs. Expected Cycle Time

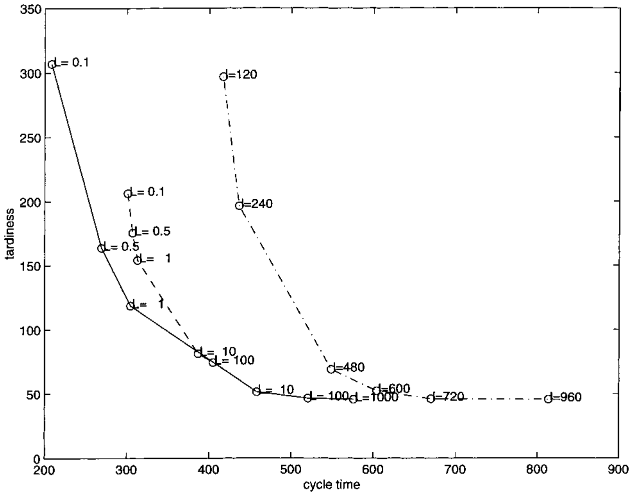


Table 2 Output of the Algorithm

Iter.	$\hat{g}_N(r^k)$	Δ	T	p-Value	N	New
1	10376.90	1284.72	Inf	0.00	50	*
2	12028.52	848.97	Inf	0.00	200	*
3	11750.88	832.37	Inf		200	
4	11478.12	792.98	Inf		200	
5	11341.37	769.28	Inf		200	
6	11237.04	742.30	Inf		200	
7	11119.22	729.78	328.56		200	
8	11045.54	696.98	Inf		200	
9	10986.39	691.41	8.10		200	
10	10935.73	681.11	80.61		200	
11	10906.81	671.74	4.91		200	
12	10898.20	671.33	63.56		200	
13	10612.28	326.79	3.85	1.00	800	*

expected tardiness and less expected cycle time than the ones given by that lead-time.

The third curve plotted on Figure 2 (the dashed line) depicts the expected tardiness and expected cycle time corresponding to the release times obtained by solving the deterministic problem in which all random variables are replaced by their means. As seen in §5, that solution can be obtained by solving a Linear Programming problem. The graph illustrates the advantage of using the stochastic optimization approach over that “naive” method: for small values of λ (i.e., bigger weight on cycle time), the solution provided by our technique yields less expected cycle time than the solution obtained with means; as λ gets large (i.e., tardiness becomes more important) the optimal “stochastic” solution results in less expected tardiness than the one with deterministic approach. Incidentally, we can also infer from the graph that, for this example problem, the deterministic approach gives better results than the lead-times policy.

We illustrate the algorithm by showing the results obtained for $\lambda = 1.0$. Table 2 shows, for each iteration k: the value of the estimator $\hat{g}_N(r^k)$, the half-size Δ of a 95% confidence interval for $g(r^k)$, the value of the statistic T described in §6 with the respective p-value and the sample size used. The last column displays an asterisk for all iterations in which a new sample was generated. Note that the p-value is not displayed for the iterations without new sample, since the lack of independence between the iterates invalidates the test.

A few words about these results. Observe that the sample size used in the first iteration (50) was actually too small, resulting in a poor estimation of the value of the function, which is reflected in the large confidence interval obtained in that iteration. The use of a bigger sample (from the second iteration on) corrected the problem, therefore causing an apparent “worsening” in the value of the function. Note also that the stopping of the program was determined by the p-value computed on iteration 13 (see §6). The obtained p-value (1.0) indicates that the corresponding solution can be accepted as optimal (i.e., the hypothesis “r^{13} is optimal” is not rejected) with a level of significance approximately equal to one, which is a strong evidence of optimality. It must be said, however, that such situation is not typical, and in fact in some other problems for which we tested the method the stopping of the program was determined by the detection of insignificance in the reduction of the value of the objective function rather than by the corresponding p-value.

We also computed the lower bounds described in §5. The first one, suggested by Jensen’s inequality (5.1), was obtained by replacing the service times with their means, and then by solving the corresponding LP problem (5.5). The resulting value was 6012. Note that this value is significantly smaller than the corresponding minimum of the expected value function, which is estimated as 10612 (see Table 2). This again illustrates the difference between solving the stochas-

Table 3 Optimal Release Times (in minutes) and Tardiness Probabilities

Job	Release Time	Due Date	Probability of Tardiness
1	10	480	0.02
2	39	480	0.04
3	93	480	0.06
4	183	480	0.11
5	205	480	0.18
6	234	480	0.24
7	305	480	0.38
8	342	480	0.58
9	379	480	0.93
10	412	480	1.00
11	508	960	0.08
12	524	960	0.11
13	596	960	0.15
14	658	960	0.21
15	712	960	0.30
16	743	960	0.38
17	792	960	0.47
18	822	960	0.62
19	858	960	0.96
20	899	960	1.00
21	947	960	1.00
22	983	960	1.00
23	1051	960	1.00
24	1100	960	1.00
25	1154	960	1.00

tic problem (3.2) and solving its deterministic counterpart (5.2). The second bound was computed by solving the optimization problem (5.6) for each sample path, and then by averaging the solutions (see (5.7)). The obtained value was 8255, with a half-size of a 95% confidence interval equal to 442 (a sample of size 200 was used). Observe that inequality (5.8) is verified here.

Table 3 shows the optimal release times together with the respective due dates and tardiness probabilities computed by (7.3). The time unit is a minute.

9. Conclusions

The model presented here represents a first step in using simulation-based optimization for production scheduling. Although the computations are intensive, the advent of manufacturing execution systems has made the data required and the platform needed for real-time execution a reality. This paper has demonstrated the feasibility of using simulation-based optimization for a small manufacturing example. Further research is needed to make this a reality for actual industrial instances.[1]

[1] The authors thank two anonymous referees and the departmental editor for their helpful comments and suggestions. This work was supported, in part, by grant Grant DMI-9713878 from the National Science Foundation. Tito Homem-de-Mello was supported by CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico), Brasília, Brazil, through a Doctoral Fellowship under grant 200595/93-8. The work of Alexander Shapiro and Mark L. Spearman was supported, in part, by grant Grant DMI-9713878 from the National Science Foundation.

References

Ahuja, R. K., T. L. Magnanti, J. B. Orlin. 1993. *Network Flows*. Prentice-Hall, New York.

Bazaraa, M. S., H. D. Sherali, C. M. Shetty. 1993. *Nonlinear Programming: Theory and Algorithms*. John Wiley & Sons, New York.

Danskin, J. M. 1967. *The Theory of Max-Min and Its Applications to Weapons Allocation Problems*. Springer, New York.

Fiacco, A. V. 1983. *Introduction to Sensitivity and Stability Analysis in Nonlinear Programming*. Academic Press, New York.

Graves, S. C. 1986. A tactical planning model for a job shop. *Oper. Res.* **34**(4) 522–533.

——, A. H. G. Rinnooy Kan, P. H. Zipkin, Eds. 1993. *Logistics of Production and Inventory*, *Vol.* 4 of *Handbooks in Operations Research and Management Science*. North-Holland, Amsterdam, The Netherlands.

Gumaer, R. 1996. Beyond ERP and MRP II: optimized planning and synchronized manufacturing. *IIE Solutions* 32–35.

Hasan, C. N., M. L. Spearman. 1996. Determining optimal material release times in stochastic production environments using infinitesimal perturbation analysis. Preprint.

Hopp, W. J., M. L. Spearman. 1996. *Factory Physics: Foundations of Manufacturing Management*. Richard D. Irwin, Chicago, IL.

Industrial Engineering. 1991. Competition in manufacturing leads to MRP II. **23** July 10–13.

Ioffe, A. D., V. M. Tihomirov. 1979. *Theory of Extremal Problems*. North-Holland, Amsterdam, The Netherlands.

Karmarkar, U. S. 1987. Lot sizes, lead times and in-process inventories. *Management Sci.* **33**(3) 409–423.

——, S. Kekre, S. Freeman. 1985. Lot-sizing and lead-time performance in a manufacturing cell. *Interfaces* **15**(2) 1–9.

Karni, R. 1982. Capacity requirements planning—a systematization. *Internat. J. Prod. Res.* **20**(6) 715–739.

Kushner, H. J., D. S. Clark. 1978. *Stochastic Approximation Methods for Constrained and Unconstrained Systems*. Springer-Verlag, New York.

Plambeck, E. L., B. R. Fu, S. M. Robinson, R. Suri. 1993. Throughput optimization in tandem production lines via nonsmooth programming. J. M. Schoen, Ed., *Proc*. 1993 *Summer Computer Simulation Conf*., Society for Computer Simulation, San Diego, CA, 70–75.

—— 1996. Sample-path optimization of convex stochastic performance functions. *Math*. *Programming* **75**(2) 137–176.

Rockafellar, R. T. 1970. *Convex Analysis*. Princeton University Press, Princeton, NJ.

——. 1968. Integrals which are convex functionals. *Pacific J. Math*. **24** 867–873.

——, R. J.-B. Wets. 1982. On the interchange of subdifferentiation and conditional expectation for convex functionals. *Stochastics* **7** 173–182.

Rubinstein, R. Y., A. Shapiro. 1993. *Discrete Event Systems*: *Sensitivity Analysis and Stochastic Optimization by the Score Function Method*. John Wiley & Sons, New York.

Saboo, S., L. Wang, W. E. Wilheim. 1989. Recursion models for describing and managing the transient flow of materials in generalized flowlines. *Management Sci*. **35**(6) 722–742.

Shapiro, A., T. Homem-de-Mello. 1998. A simulation-based approach to stochastic programming with recourse. *Math*. *Programming* **81**(3) 301–325.

Spearman, M. L., M. A. Zazanis. 1992. Push and pull production systems: issues and comparisons. *Oper*. *Res*. **40** 521–532.

Vollmann, T. E., W. L. Berry, D. C. Whybark. 1988. *Manufacturing Planning and Control Systems*. 2nd ed. Richard D. Irwin, Homewood, IL.

Accepted by Pierre L'Ecuyer; received November 27, 1996. *This paper has been with the authors* 8 *months for* 3 *revisions*.