A price-directed heuristic for the economic lot scheduling problem

DANIEL ADELMAN¹ and CHRISTIANE BARZ²∗

¹University of Chicago, Booth School of Business, Chicago, IL 60637, USA
²University of California, Anderson School of Management, Los Angeles, CA 90095, USA
E-mail: christiane.barz@anderson.ucla.edu

Received March 2012 and accepted February 2014

The article formulates the well-known economic lot scheduling problem (ELSP) with sequence-dependent setup times and costs as a semi-Markov decision process. Using an affine approximation of the bias function, a semi-infinite linear program is obtained and a lower bound for the minimum average total cost rate is determined. The solution of this problem is directly used in a price-directed, dynamic heuristic to determine a good cyclic schedule. As the state space of the ELSP is non-trivial for the multi-product setting with setup times, the authors further illustrate how a lookahead version of the price-directed, dynamic heuristic can be used to construct and dynamically improve an approximation of the state space. Numerical results show that the resulting heuristic performs competitively with one reported in the literature.

Keywords: Economic lot scheduling problem, sequence-dependent setups, semi-Markov decision process, approximate dynamic programming, dynamic heuristic

1. Introduction

The classic economic lot scheduling problem (ELSP) schedules production runs for several products on a single machine. Demand and production rates are constant. A setup (of a given cost and time) must be performed every time production is changed from one product to another. The aim is to minimize the long-run time-average sum of holding and setup cost over an infinite horizon without incurring any stock-outs.

In this article, we use an approximate dynamic programming (ADP) framework to develop a heuristic to construct a cyclic schedule for the sequence-dependent ELSP. The resulting heuristic is the main contribution of this article as a new competitive alternative to existing approaches. A second contribution is to ADP methodology, as we provide a novel approach to iteratively approximating an implicitly defined state-space through policy simulation. In both cases, our contributions open avenues for future work exploring stronger heuristics for the ELSP and developing simulation-based state-space approximations in ADP.

The literature on the ELSP has focused on cyclic schedules, which repeat a given production cycle indefinitely. Many authors restrict attention to cycles having particular forms, such as rotation or power-of-two schedules. Typically, schedules are constructed by first solving a math program that approximates optimal production frequencies, and then heuristically adjusting them. In related work (Adelman and Barz, 2014), we formally derived these math programs by making an affine value function approximation to the dynamic programming formulation of the ELSP with sequence-dependent setup times and costs. Such a formulation does not restrict itself to cyclic schedules.

In this article, we use the resulting approximate dual prices in a dynamic heuristic to obtain an intuitive economic control mechanism. Unlike previously published heuristics, this approach uses the parameters obtained from the corresponding math problem directly to determine production and idling times and allows for an economic interpretation of each parameter used. The production decision reduces to a problem of finding a good balance between generating product value and avoiding machine usage and setup cost. Repeatedly solving for the next production action to take given the current inventory state on a rolling horizon basis, we obtain a cyclic schedule if a state is revisited; otherwise, we terminate the procedure after a certain number of steps and use the generated sequence as a production sequence. As a post-processing step, in either case, we optimize production and idling times, given the resulting cyclic production sequence. To our knowledge, ours is the first dynamic approach to constructing cyclic schedules.
to be reported in the literature. Although, in general, our heuristic can be implemented dynamically as the inventory state evolves, we focus on converting dynamically generated production sequences into repeatable cyclic schedules to allow for a comparison of our heuristic with other methods in numerical examples.

We address a significant technical difficulty, arising from the need to ensure that no stockouts occur: We begin the heuristic with a subset of the state-space for which we know a production schedule that avoids stockouts forever. As the horizon rolls through time, we dynamically expand this state-space approximation to include more states as they are visited. We show theoretically that these additional states can be added without inducing stockouts in the future. These new states open up the possible actions explorable by the dynamic heuristic as it evolves through time.

1.1. Related literature

Using price-directed control for the ELSP is a new way of viewing the production/scheduling decisions in this problem. While most of the research on the (deterministic) ELSP has focused on the construction of cyclic schedules and has ignored sequence-dependent setups, dynamic approaches are more common for its stochastic counterpart, the stochastic economic lot scheduling problem (SELSP), which allows for backlogging of unsatisfied demand.

1.1.1. The sequence-dependent ELSP

Ever since the ELSP was first introduced by Rogers (1958), a large number of heuristics, among them the so-called common-cycle, basic-period, and varying-lot-sizes approaches, have been proposed (see, e.g., the reviews given in Elmaghraby, 1978; Davis, 1995; and Carstensen, 1999). Although sequence-dependent setup times and costs are reported to be prevalent in most practical applications (see, e.g., Monkman et al., 2008 and Mehtrotra et al., 2011), the vast majority of the ELSP-related research concentrates on the special case of sequence-independent setup times and costs. Early exceptions like Maxwell (1964), Sing and Foster (1987), and Inman and Jones (1993) considered very restrictive forms of setup times. One of the most sophisticated algorithms for sequence-dependent setups was suggested by Dobson (1992). He developed a lower bound problem for the optimal average total cost rate to obtain the production frequency of each product. Referring to Roundy’s (1989) results on the guaranteed performance of power-of-two-heuristics given sequence-independent setups, Dobson constructed a cyclic production schedule with power-of-two-frequencies. Taylor et al. (1997) developed a simple heuristic, based on economic order quantity frequencies and a production sequence, which is obtained by applying a travelling salesman algorithm. Wagner and Davis (2002) proposed a search heuristic over the set of cyclic production schedules. For a general overview on scheduling and lot-sizing with sequence-dependent setups see Zhu and Wilhelm (2006), Jans and Degraeve (2007) review meta-heuristics for lot-sizing.

To our knowledge, there is no literature about scheduling production runs in the sequence-dependent ELSP that considers dynamic policies. Aragone and Gonzalez (2000) analyzed a dynamic policy for sequence-independent problems with negligible setup times. Goncalves and Leachman (1998) and Segerstedt (1999) proposed dynamic heuristics over a finite horizon; Gallego (1990b, 1994), Eisenstein (2005), and Robinson and Chen (2008) suggested methods to recover from a disrupted cyclic (target) schedule, allowing backordering when demand cannot be met instantaneously. The ELSP over cyclic schedules was extended to allow for backlogging of demand or lost sales by Gallego and Roundy (1992), Gupta (1992), Altiiok and Shieue (1994), and Faaland et al. (2004).

Dynamic (infinite-horizon) policies were mainly suggested for the stochastic version of the ELSP, the SELSP, see Sox et al. (1999) and Winands et al. (2005) for overviews. In such a stochastic environment, different semi-Markov formulations of the problem with backlogging of unsatisfied demand were proposed by Qui and Loulou (1995) and Hodgson et al. (1997). Due to the curse of dimensionality, however, they only solve problems with a very small number of products. Qui and Loulou (1995) used a discounted-cost criterion and Poisson arrivals of demand, allowing the decision-maker to switch between products or to idle every time the production of an item finishes (or when demand arrives while the machine was idling). They restricted their experiments to settings with only two products. Hodgson et al. (1997) explicitly focused their attention on problems with at most four identical products.

The ELSP has been shown to be NP-hard under cyclic schedules (see Hsu, 1983 and Gallego and Shaw, 1997). A problem related to the ELSP is the question of whether given starting inventory levels are sufficient to serve demand over an infinite horizon without stock-outs. Anderson (1990) showed that this lot scheduling feasibility problem also is NP-hard.

Adelman and Barz (2014) showed how an affine approximation of a semi-Markov decision formulation of the ELSP unifies previously obtained lower bound results and suggested the intuitive interpretation of their approximation parameters. They, however, did not study how to use their approximation in a price-directed heuristic. This article will close this gap.

1.1.2. Price-directed control and approximate dynamic programming

Our heuristic is explicitly derived from an affine approximation of the value function of a semi-Markov decision process. The idea of price-directed control has been used in a variety of heuristics; e.g., to control remnant inventory systems by Adelman and Nemhauser (1999), in revenue management by Talluri and van Ryzin (1998), and...
in vehicle dispatching by Gans and van Ryzin (1999). Adelman (2004a, 2004b) also derived their heuristics from a dynamic programming formulation of a joint replenishment and inventory/routing problem. As in Adelman and Barz (2014), we determine prices by the solution of a relatively small convex optimization problem.

Overviews on ADP are given by Bertsekas and Tsitsiklis (1996), Bertsekas (2005), and Powell (2007). Like much of the literature on ADP, their focus lies on simulation-based methods for adaptively computing value function approximations. Our approximations will lead to an optimization problem that can be solved directly with no need for simulation.

1.2. Outline

After a short introduction to the semi-Markov decision process formulation of the sequence-dependent ELSP, we consider the affine approximation suggested by Adelman and Barz (2014) in Section 2. This approximation leads to a small convex quadratically constrained problem, which we will use to determine dual prices of inventory, setup times, and machine time. In Section 3, we apply the idea of price-directed control in a greedy fashion. In Section 4, we discuss how the dynamic heuristic can be used to construct a cyclic schedule. Section 5 presents numerical experiments. Section 6 concludes and highlights directions for future research.

2. The economic lot scheduling problem

In the classic formulation of the ELSP, one machine can be used to produce \( I \) different products \( i = 1, \ldots, I \), \( I \in \mathbb{N} \). We denote the set of products as \( \mathcal{I} = \{1, \ldots, I\} \). Only one product can be produced at a time. When switching from one product \( i \) to another product \( j \), a setup must be performed, incurring cost \( 0 \leq \tau_{ij} < \infty \) and requiring a setup time of \( 0 \leq \tau_{ij} < \infty \) during which production is paused. Product \( i \) is produced at rate \( \lambda_i \) and requiring \( \lambda_i \) items per unit time. The demand rate is constant at \( \lambda_i \) items per unit time. For each product \( i \in \mathcal{I} \), \( 0 \leq \lambda_i < \rho_i \). The utilization for product \( i \) is denoted by \( \rho_i = \lambda_i / \rho_i \). We assume \( \sum_{i=1}^{I} \rho_i < 1 \).

Finished products of type \( i \) that are not yet consumed by demand incur holding cost of \( \phi_i > 0 \) per time unit and item. We do not allow for stock-outs because, as discussed in Section 6, a number of methodological issues arise in this case.

2.1. A semi-Markov decision problem

We formulate the semi-Markov decision problem as proposed in Adelman and Barz (2014). The controller wants to minimize long-term average total cost from a given system state \((s^1, m^1)\), which describes the initial machine state \( m^1 \) (the product for which the machine is setup) and the initial inventory levels for all products \( s^1 = (s_1, \ldots, s_I) \).

Let \((s^n, m^n)\) denote the state in decision epoch \( n \). The controller then chooses an action composed of an idle time \( u^n \geq 0 \), the index of the next product the machine should produce \( q^n \in \mathcal{I} \), and the corresponding production time \( t^n \geq 0 \). This action \((u^n, q^n, t^n)\) is chosen such that no stock-outs occur. Then, in epoch \( n + 1 \), the new machine state is

\[
m^{n+1} = q^n \quad \text{for all } n \in \mathbb{N},
\]

and the new inventory levels after idling time \( u^n \), setup time of \( \tau_{nq^n} \) for product \( q^n \), and production time \( t^n \) are \( s^{n+1} = (s_1, \ldots, s_I) \).

Using \( \delta(q^n = i) = 1 \) if \( q^n = i \) and zero else. To avoid stock-outs, it must hold that \( s_i^n - \lambda_i u^n + \tau_{nq^n} t^n \geq 0 \) and \( s_i^{n+1} \geq 0 \) for all \( i \in \mathcal{I} \).

All holding costs for items produced during time \( t^n \) (which we will call the \( n \)th production run) are incurred in decision epoch \( n \). Hence, the incremental holding cost from this production run equals:

\[
k(q^n, m^n, u^n, q^n, t^n) = c_{mr^q} + \frac{\phi_{qr^q} p_{qr^q}(t^n)^2 (p_{qr^q} - \lambda_{qr^q})}{2\lambda_{qr^q}} + \frac{\phi_{qr^q} p_{qr^q} t^n (s_{qr^q} - \lambda_{qr^q} (u^n + \tau_{mr^q} t^n))}{\lambda_{qr^q}},
\]

where the first term represents the switching cost, the second the inventory costs caused by new items that will be produced during time \( t^n \) (the light gray shaded area in Fig. 1), and the last term adds extra inventory holding costs that are incurred when production is started before the inventory level has reached zero (the dark gray shaded area in the same figure).

![Fig. 1. Illustration of holding cost. (To simplify the notation, the superscript \( n \) is suppressed for all variables.)](image-url)
Further, let \( \pi_{(s,m)} = \{(s^n, m^n), (u^n, q^n, t^n)\}_{n=1,2,...} \) denote an infinite sequence of state–action pairs fulfilling Equations (1) and (2), \( s^n, u^n, t^n \geq 0, \) and \( q^n \in \mathcal{S} \) for all \( n \in \mathbb{N}, i \in \mathcal{I} \) with initial inventory state \( s^1 = s \) and machine state \( m^1 = m \). The average total cost of a trajectory, \( \pi_{(s,m)} \), is

\[
J(\pi_{(s,m)}) = \limsup_{N \to \infty} \frac{\sum_{n=1}^{N} k(s^n, m^n, u^n, q^n, t^n)}{\sum_{n=1}^{N} (u^n + \tau m q^n + t^n)}.
\]

Using this notation, we can formulate the decision-maker’s problem to find an optimal trajectory \( \pi_{(s,m)} \) as

\[
\inf_{\pi_{(s,m)}} J(\pi_{(s,m)}).
\]

The state space \( \mathcal{S} \) is chosen to be the set of all combinations of machine states \( m \in \mathcal{I} \) and initial inventories \( s \in \mathbb{R}_+^I \) for which there exists a trajectory that avoids stock-outs; i.e., for which all products \( i \in \mathcal{I} \) and decision epochs \( n \in \mathbb{N} \) the inventory level after idling and setup (when it is lowest) is non-negative:

\[
s^n_i - \lambda_i \times (u^n + \tau m q^n) \geq 0 \quad \text{for all} \quad i \in \mathcal{I}, \quad n \in \mathbb{N}.
\]

Given machine state \( m \), we denote the set of all inventory levels that form a feasible state by

\[
\mathcal{S}(m) = \{ s \mid (s, m) \in \mathcal{S} \}
\]

and call an inventory level \( s \) feasible in machine state \( m \) if \( s \in \mathcal{S}(m) \). The set \( \mathcal{S} \) is non-empty if \( \sum_{i=1}^{I} \rho_i < 1 \). In general, however, the structure of \( \mathcal{S} \) is not trivial. Anderson (1990) showed that even in the case of sequence-independent setup times, it is NP-hard to decide whether a given state \( (s, m) \) lies in \( \mathcal{S} \). Furthermore, the sets \( \mathcal{S}(m) \) need, in general, to be neither convex nor closed as Adelman and Barz (2014) show.

We define the set of feasible actions \( \mathcal{A}(s,m) \) when starting in \( (s,m) \) as the set of all actions \( (u,q,t) \) that prevent a direct stock-out by fulfilling Equation (4) and that also prevent long-term stock-outs by leading to another state in the state space \( \mathcal{S} \). To make this exact, let \( s'(s,m,u,q,t) \) represent the inventory vector after action \( (u,q,t) \) was executed, starting from state \( (s,m) \); i.e., \( s'(s,m,u,q,t) = s_i - \lambda_i \times (u + \tau m q) + \delta(q = i) p q t \) for all \( i \in \mathcal{I} \). Then,

\[
\mathcal{A}(s,m) = \{(u,q,t) : t + u \geq 0, q \in \mathcal{S}, s'(s,m,u,q,t) \in \mathcal{S}(q), s_q - \lambda_q \times (u + \tau m q) \geq 0\}.
\]

Since \( s'(s,m,u,q,t) \in \mathcal{S}(q) \) ensures the non-negativity of inventory before production; i.e., after time \( u + \tau m q \), for all \( i \neq q \), we need to postulate this property only for product \( q \).

We denote the set of all feasible idling and production times \( (u,t) \) given \( q \) by

\[
\mathcal{A}(s,m,q) = \{(u,t) : u \geq 0 : (u,q,t) \in \mathcal{A}(s,m)\}.
\]

Since the state-space has a rather non-trivial structure, the optimal average total cost (rate) might depend on the system state, which we denote by \( g(s,m) \). The difference between the optimal total cost for a system starting in state \( (s,m) \) and the optimal cost accumulated at the stationary rate \( g(s,m) \) over the infinite horizon is captured by the bias function (see, e.g., Bertsekas, 2007). Denoting the bias function by \( h(s,m) \) and taking into account that given action \( (u,q,t) \), the decision epoch lasts \( u + \tau m q + t \) time units, it is straightforward to write the optimality equations as

\[
g(s,m) = \inf_{(u,q,t) \in \mathcal{A}(s,m)} g(s'(s,m,u,q,t), q) \quad \text{(5)}
\]

\[
h(s,m) = \inf_{(u,q,t) \in \mathcal{A}(s,m)} \{k(s_q, m, u, q, t) - g(s,m) \times (u + \tau m q + t) + h(s'(s,m,u,q,t), q)\} \quad \text{(6)}
\]

for all \( (s,m) \in \mathcal{S} \), where \( \mathcal{A}_q(s,m) \) is the set of actions \( a \in \mathcal{A}(s,m) \) that minimize the right-hand side of Equation (5); see Schäl (1992). Even if the existence of a solution were ensured, however, the above equations would be intractable due to the continuous nature of the state and action spaces.

### 2.2. Cyclic schedules

Most of the solution methods for the ELSP have assumed that the controller’s actions should follow a certain production cycle, \( ((u^{[n]}, q^{[n]}, t^{[n]})\}_{n=1,\ldots,N} \) with \( I \leq N < \infty \), and repeat this cycle infinitely; i.e., \( u^{[n+\theta N]} = u^{[n]} \), \( q^{[n+\theta N]} = q^{[n]} \), and \( t^{[n+\theta N]} = t^{[n]} \) for all \( n = 1, \ldots, N \) and \( \theta \in \mathbb{N} \). In this case, we call \( ((u^{[\cdot]}, q^{[\cdot]}, t^{[\cdot]})\}_{n=1,\ldots,N} \) a cyclic schedule of length \( N \). To allow for an infinite number of repetitions, the parameters of the cyclic schedule are typically chosen such that inventory levels of each product are the same at the beginning and the end of each cycle. To simplify notation, we refer to the production sequence of such a cyclic schedule as \( q = (q^{[1]}, \ldots, q^{[\theta N]}) \). For given \( q \), idling and production times that minimize the average setup and holding cost can be determined; for example, by solving the nonlinear problem outlined in Online Appendix B. The difficulty lies in the determination of a good production sequence \( q \). Cyclic schedules of length \( N = I \) are commonly referred to as rotation schedules. Jones and Inman (1989) and Gallego (1990a) derive conditions under which such rotation schedules are near optimal.

### 2.3. A semi-infinite linear program formulation

In our semi-Markov decision problem formulation (5)–(6), we do not constrain ourselves to schedules of any particular structure. To make it tractable, Adelman and Barz (2014) introduce variables \( V_1, \ldots, V_I, \theta_1, \ldots, \theta_I \), and \( \xi \), to parameterize the average total cost \( g(s,m) \) and approximate the
bias function $h(s, m)$ as
\[
g(s, m) = \sum_{i=1}^{I} \lambda_i V_i - \xi, \quad \forall (s, m) \in S, \tag{7}
\]
\[
h(s, m) \approx \theta_m - \sum_{i=1}^{I} s_i V_i, \quad \forall (s, m) \in S. \tag{8}
\]

Adelman and Barz (2014) show that the semi-infinite linear program obtained by plugging Equations (7) and (8) into the linear programming formulation of the Optimality Equations, (5) and (6), gives a lower bound on the average total cost of all trajectories that keep the inventory levels within a bounded set and move forward in time. The dual of this semi-infinite linear program inspires an intuitive interpretation of the approximation parameters. $V_i$ is the dual price of the constraint that the demand rate of product $i$ must equal its time-averaged production rate. Parameter $\theta_i$ is the dual price of the constraint that the rate at which changeovers to product $i$ occur must equal the rate at which changeovers from product $i$ are made. Finally, $\xi$ is the dual price of the capacity restriction given a single machine.

With this interpretation, the above parametrization of the average total cost (7) can be viewed as the difference of the marginal value gained from serving demand per time unit (at rate $\lambda_i$) demand is satisfied with products of marginal value $V_i$ per unit) and the marginal value of machine time, $\xi$. The bias function (8) is composed of a marginal value of being in machine state $m$, $\theta_m$, and the value of all products on inventory (all $s_i$ items on inventory are weighted by their marginal values $V_i$). The value of all products in inventory is subtracted because high inventories of products allow the costs of production and holding the produced items to be delayed. Hence, the difference between the total cost, starting in $(s, m)$, and the stationary cost $g(s, m)$ over the infinite horizon should be decreasing in the inventory level of each product $i \in \mathcal{I}$.

Denoting by $\tau_{pq}$ the length of the shortest setup time path from product $q$ back to product $p$, Adelman and Barz (2014) further show the following results, which we summarize in Theorem 1.

**Theorem 1.** The convex program:
\[
\begin{align*}
\max_{\xi \geq 0, V_i, \theta, \theta_q, \theta_m = \text{const}} & \sum_{i=1}^{I} \lambda_i V_i - \xi, \\
\text{subject to} & \theta_m - \theta_q - \xi \tau_{mq} - c_{mq} + \frac{(p_q V_q - \xi)^2}{2\phi_q (p_q - \lambda_q)} \leq 0, \forall m, q \in \mathcal{I}, \\
& \text{(a) has the same solution and objective value as the semi-} \\
& \text{infinite linear program if } \tau_{pq} \leq \tau_{mq} (1 - \rho_q)/(1 - \sum_{i=1}^{I} \rho_i) \\
& \text{for all } m, q \in \mathcal{I}, \text{ (b) gives a lower bound to the semi-infinite linear program otherwise, and (c) has the same value as the lower bound problem given in Dobson (1992)}.
\end{align*}
\]}

**Table 1.** Maximum, minimum, and average solution CPU times of (9)–(10) in seconds

|  | 10 | 25 | 50 | 100 | 200 | 250 |
|---|---|---|---|----|----|----|
| Average | 0.03 | 0.38 | 2.58 | 29.17 | 346.34 | 810.61 |
| Minimum | 0.02 | 0.34 | 2.25 | 26.65 | 321.82 | 700.56 |
| Maximum | 0.04 | 0.42 | 2.95 | 34.45 | 378.33 | 916.86 |

Many general-purpose solvers can handle convex problems such as (9)–(10) well. For problems of size $I = 10, 25, 50, 100, 200$, and $250$, we constructed 20 random problem instances, each mimicking the structure of Dobson’s (1992) simulation setup choosing $p, s$ uniform in $[100, 400]$; $\lambda$, uniform to obtain a utilization of 0.9; holding cost $\phi$ uniform in $[0.01, 0.03]$; parameters $c_i$ and $\tau_i$ uniform in $[0.1, 0.3]$ and $[0.3, 0.9]$; and setup costs (times) $\tau_{ij}(\tau_{ij})$ uniform in $[c_i, 3c_i]$ for all $i, j \in \mathcal{I}$. Maximum, minimum, and average solution times of a straightforward implementation of Program (9)–(10) in AMPL using MINOS on a Linux workstation having an Intel Xeon 3.6GHz CPU are reported in Table 1. Although the solution times increase exponentially in $I$, problems of realistic size can be solved within seconds.

### 3. A price-directed dynamic heuristic

The prices obtained from solving Program (9)–(10) can be used in a price-directed, dynamic heuristic. The price-directed heuristic follows the dynamic programming idea of minimizing the right-hand side of the Optimality Equation (6) to find good actions, given inventories $s$ and machine state $m$. Using our approximation of the bias function and multiplying the optimality equation by $(-1)$, this translates into finding the feasible action $(u, q, t)$ that solves:
\[
\begin{align*}
\max_{u, q \in \mathcal{I}, t \geq 0} & \quad tp_q V_q - \xi^*(u + \tau_{mq} + t) - (\theta_q^* - \theta_m^*) \\
& \quad - \kappa(s_q, m, u, q, t), \\
& \quad s_q - \lambda_q(u + \tau_{mq}) \geq 0, \\
& \quad s - \lambda(t_{mq} + t + u) + \rho e_t \in \mathcal{S}(q).
\end{align*}
\]}

We break ties between two solutions with different $q$-values by choosing the solution that sets up the product with smallest $\rho$-value of using the machine; it is the machine cost $\tau_{pq}$ multiplied by the length of the decision epoch, $(u + \tau_{mq} + t)$. The third summand is the loss in setup value, and $\kappa(s_q, m, u, q, t)$ is the actual total cost incurred, the sum of setup and
inventory cost. This problem can hence be interpreted as a net-value maximization over all feasible actions \((u, q, t)\) in state \((s, m)\). Unfortunately, the determination of feasible actions \((u, q, t)\) given state \((s, m)\) is difficult in general since there is no known closed-form representation of \(S\).

### 3.1. The case of zero setup times

In the online appendix, we show that in the one-product case, the price-directed heuristic prescribes the well-known optimal policy, as discussed in Zipkin (2000). For more than one product, our heuristic is not guaranteed to produce the optimal policy. Even the state and action spaces might be difficult to characterize.

In the case of zero setup times, however, the state space can be described by \(S = \{(s_1, \ldots, s_I, m) \in (R^I)^I \times \mathbb{I} : s_i + s_j > 0 \ \forall i \neq j\}\). If at most one inventory position is zero, production can be switched to the product with lowest inventory and continued until another product is in danger of stocking out. Because only one product can be produced at a time, states in which two or more products have zero inventory are infeasible. Remember that feasible actions have non-negative production and idling times and lead to a feasible state.

In the following, we present a small two-product example demonstrating the core idea of our price-directed heuristic. Take \(\lambda_1 = \lambda_2 = 1, p_1 = p_2 = 8, \phi_1 = 1, \phi_2 = 9, \tau_{ij} = 0\) for all \(i, j = 1, 2, c_{11} = c_{21} = 4, c_{12} = 2, c_{22} = 1\). Because \(I = 2\) and setup times are zero, the state space is \(S = \{(s_1, s_2, m) \in \mathbb{R^+} \times \mathbb{R^+} \times \{1, 2\} : (s_1, s_2) \neq (0, 0)\}\).

Solving Program (9)-(10) gives a lower bound of 6.614 with \(V_1^* = 2.646, V_2^* = 3.969, \xi^* = 0, \text{ and } \theta_1 = -0.365\).

(Remember that we set \(\theta_2 = 0\) in Program (9)-(10).)

Starting in state \((2.201, 0.236, 1)\), which is marked by a cross in both Figs. 2 and 3, and maximizing the net value (11)-(13) for \(s^1 = (2.201, 0.236)\) and \(m^1 = 1\) gives \((u, q, t) = (0.236, 2, 0.062)\), leading to \(s^2 = (1.903, 0.441)\) and \(m^2 = q = 2\). The resulting inventory trajectory is depicted as a dashed line and labeled as step I in Fig. 2. Because only product 2 is produced in step I, the inventory of product 1 decreases throughout. In state \((1.903, 0.441, 2)\), we can again maximize the net value (11)-(13) to obtain \((u, q, t) = (0.441, 2, 0.062)\) leading to \(s^3 = (1.400, 0.441)\) and \(m^3 = q = 2\), labeled as step II. Continuing this approach gives steps III–V. Further continuation results in the full trajectory depicted as a solid black line in Fig. 2. In the long run, the heuristic follows the same cyclic schedule of length 7 over and over again. The average total cost incurred by repeating this schedule indefinitely is 6.937, only 4.9% higher than our lower bound.

For this example, the heuristic suggested by Dobson (1992) yields a cyclic schedule of length 5 with average total cost of 7.155 per time unit, 8.2% higher than the lower bound. Figure 3 depicts the inventory trajectory of that cyclic schedule.

![Fig. 2. Inventory trajectory when following the price-directed heuristic.](image)

![Fig. 3. Inventory trajectory when following the cyclic schedule found by Dobson's algorithm.](image)
from solving a Lower-bound problem. Unsurprisingly, this calculation tends to underestimate the marginal values of the products and produce less than the optimal amount of a product. In the case in which the lower bound is loose, this underestimation can result in insufficient production quantities and too frequent setups (compared to the optimal schedule).

### 3.2. Multiple products and setup times

If setup times are zero or if there is only one product, both the state and action spaces are relatively easy to characterize. In general, however, Condition (13), which ensures feasibility of an action, seems to be difficult to satisfy. This is why we will suggest an approximation procedure for the action space in instances with more than one product and positive setup times. In this section, we introduce a state-space approximation based on feasible states that are easy to obtain. In the next section, we will show how to adaptively expand this approximation over time.

The main idea behind the approximation is the following: If we know a feasible inventory vector \( \bar{s}(m) \in S(m) \), we can conclude that stock-outs can also be avoided starting with higher inventories: all inventory vectors \( s \) with \( s_i \geq \bar{s}_i(m) \) for all \( i \in I \) are in \( S(m) \). So, given a collection of feasible inventory levels \( S^W(m) = (\bar{s}(m), \ldots, \bar{s}(m)) \) for each \( m \in I \) with \( \bar{s}(m) \in S(m) \) for all \( w = 1, \ldots, W_n \), we can solve:

\[
\begin{align*}
\max_{u \geq 0, q \in I, t \geq 0, w \in \{1, \ldots, W_n\}} & \quad \theta_m^u - \phi_q^u \mu_q^u t p_q V_q^u - \xi^u (u + \tau_mq + t) \\
- & c_mq - \frac{\phi_q p_q t}{\lambda_q} (2s_q - 2\lambda_q(u + \tau_mq) + t(p_q - \lambda_q)), \quad (14) \\
& s_\gamma_q - \lambda_q(u + \tau_mq) \geq 0, \quad (15) \\
& s_i - \lambda_i(s_\gamma + t + u) + \delta(i = q)p_i t \geq \bar{s}_i(q) \quad \text{for all } i \in I, \quad (16)
\end{align*}
\]

instead of (11)–(13).

We will denote by \( \hat{S}^W(m) = \{s : \exists w \in \{1, \ldots, W_n\} \text{ the approximation of } S(m) \text{ given inventory vectors } (\bar{s}(m), \ldots, \bar{s}(m)) \text{ and by } \hat{A}^W(m) = \{(u, q, t) : u \geq 0, q \in I, s'(s, m, u, q, t) \in \hat{S}^W(m), s - \lambda_q(u + \tau_mq) \geq 0 \text{ the corresponding approximation of } A(s, m) \text{ A visualization of an approximation of } \hat{S}^W(m) \text{ with } W = 3 \text{ is given in Fig. 4.}

Even in the presence of positive setup times, the generation of individual, feasible states is easy. Feasible states are, for example, those visited on a non-idling rotation schedule that sets up products 1 through \( I \) repeatedly and produces product \( i \in I \) for time \( t' = r_t/\bar{r} \) with \( r_t = \lambda_t/p_t \), \( \bar{r} = \sum_{i=1}^I \lambda_t/p_i \) and \( \bar{r} = r_t = \tau_{t+1} + \sum_{i=1}^I \tau_{t+1} \). So, a collection of feasible inventories with \( W_q = 1 \) for all \( q \in I \) is, for example, given by \( \tilde{s}_i'(q) = \lambda_i \tau_{q+1} + \lambda_i \sum_{j=q+1}^{I} (\tau_{j+1} + t'_j) \) for all \( q < i, \) \( \tilde{s}_i'(q) = \lambda_i \tau_{q+1} + \lambda_i \sum_{j=q+1}^{I} (\tau_{j+1} + t'_j) + \lambda_i \sum_{j=1}^{I} (\tau_{j+1} + t'_j) \) for all \( i < I \) and \( \tilde{s}_i'(I) = \lambda_i \tau_{I+1} + \lambda_i \sum_{j=1}^{I} (\tau_{j+1} + t'_j) \). We will refer to the approximation based on this collection of feasible inventory levels as the default initial (state space) approximation.

Other feasible inventory states can easily be determined from other cyclic schedules. Anderson (1990) shows in his Theorem 1 that all states in the interior of \( S \) can be constructed from cyclic schedules in the sequence-independent case. His proof fully carries over to the sequence-dependent case.

The case with zero setup times can be implemented as a special case of the above. Since at most one of the \( I \) inventory levels may be zero, we can choose \( \epsilon > 0 \) and set \( W_q = 1 \) for all \( q \in I, \) \( \tilde{s}_i'(q) = \epsilon \) for all \( j \neq i \) and \( \tilde{s}_i'(q) = 0 \) for all \( i \in I \). This approximation will get closer and closer to the true state and action space as \( \epsilon \) gets smaller. The problem we have in the case of positive setup times is that we need the set of all minimal inventory levels to obtain a good approximation and it is unclear how these minimal inventory levels can be obtained.

### 4. The dynamic construction of a cyclic schedule

Our discussion so far suggests to first approximate \( S(m) \) and then calculate the operating policy by repeatedly solving (14)–(16), given the current state \( (s, m) \). There are, however, two problems with pursuing this approach. First and foremost, it is not clear how to construct a set of minimal inventory levels that leads to a good approximation of \( S \) in the general case with positive setup times. Second, the one-step net-value maximization (14)–(16) frequently turns out to be too greedy.

A common way to solve the second problem is to maximize the net value over the next \( M \) decision epochs. This \( M \)-step lookahead problem can be formulated in a fashion similar to (14)–(16) with variables \( u \geq 0, q \in I, t \geq 0 \) for each of the \( M \) steps, constraints to ensure...
non-negative inventories at each step and an additional constraint ensuring that the final state, after the Mth step, is in the current approximation of S.

Given approximations \( \hat{S}^{M_n}(m) \) for all \( m \in \mathcal{I} \), letting \( m = q_0 \), and introducing \( s^a \) to denote the inventory position before the \( n \)th decision is made, the optimization problem for the \( M \)-step lookahead problem starting in state \((s^1, q^0)\) can be formulated as

\[
\begin{aligned}
&\max_{w, a^n, q^n, t^n, s^n, n=1, \ldots, M} \quad \theta^*_m - \theta^*_q + \sum_{n=1}^M t^n p_{q^n} V_{q^n}^a \\
&- \xi \sum_{n=1}^M (u^n + \tau_{q^n-1} q^n + t^n) - \sum_{n=1}^M c_{q^n-1} q^n \\
&- \sum_{n=1}^M \Phi_{q^n} p_{q^n} t^n - \frac{2}{\lambda q^n} (2q^n - 2\lambda q^n (u^n + \tau_{q^n-1} q^n) + t^n (p_{q^n} - \lambda q^n)),
\end{aligned}
\]

(17)

Proof. The solution of (17)–(21) has the interpretation of a state-action sequence of length \( M \) with states \((s^n, q^n)\) and actions \((a^n, q^n, t^n)\) for all \( n = 1, \ldots, M \). After the last action, the system will be in state \((s^{M+1}, q^M)\) with \( s^{M+1} = \hat{s}^M (q^M) \) as given by the left-hand side of Equation (19).

Part (a) can be shown by backwards induction. Obviously, \((s^{M+1}, q^M) \in S\) because \( s^{M+1} = \hat{s}^M (q^M) \) for all \( i \in \mathcal{I} \). Taking into account the Flow-balance Equations (20), action \((a^M, q^M, t^M)\) is feasible in state \((s^M, q^M)\) because it prevents an immediate stock-out by Equation (18) and leads to \((s^{M+1}, q^M)\), which is in S. Summarizing this, we know that there is an infinite sequence of state-action pairs that prevents stock-outs starting from \((s^{M+1}, q^M)\) and there is a feasible action that takes us from \((s^M, q^{M-1})\) to \((s^{M+1}, q^M)\). We can conclude that there exists an infinite sequence of state-action pairs that prevents stock-outs, starting from \((s^M, q^{M-1})\). Hence, \((s^M, q^{M-1}) \in S\). This argumentation can be repeated for all \( n = M - 2, \ldots, 1 \). We can conclude that \((a^{(s^n+1)}, q^n) \in S\) for all \( n = 1, \ldots, M \) and \((b)^{(a)} (a^n, q^n, t^n) \in S\) as the “middle” states and to the state \((s^n, q^n)\) wisely, there is no need to enforce that each of the \( M \) steps end within our approximation of \( S \) as the following theorem shows.

4.1. Dynamic updates of the approximation of the state and action spaces

Notice that we only enforce that the final state is in our current state space approximation. If we choose our starting set of \( \mathcal{S}^w(m) \) wisely, there is no need to enforce that each of the \( M \) steps end within our approximation of \( S \) as the following theorem shows.

Theorem 1. Assume that sets \( \hat{S}^{W_1}(1), \ldots, \hat{S}^{W_I}(I) \) as described in Theorem 1, each inventory vector \( s^n, n = 2, \ldots, M, \) of the solution of Program (17)–(21) is added to the set \( \hat{S}^{W_{q_n}}(q_n-1) \), to obtain sets \( \hat{S}^{W_1}(1), \ldots, \hat{S}^{W_I}(I) \), with \( W_1 \geq W'_1, \ldots, W_I \geq W'_I \) elements. Then, there still exists an infinite sequence of state-action pairs \((s^n, m^n)\), \((a^n, q^n, t^n)\), \( n = 1, 2, \ldots \), with \((s^n, m^n) \in \hat{S}^{W_{q_n}}(q_n) \) and \((a^n, q^n, t^n) \in \hat{A}^{W_{q_n}}(m^n) \) for each \((s^1, m^1) = (s, m) \) in the approximation \( \hat{S}^{W_{q_n}}(m) \).

Proof. From the proof of Theorem 1, we know that all inventory states \( s^{n+1} \) are in \( S(q_n) \) and that there exists an infinite sequence of state-action pairs that prevents stock-outs over the long term, starting in state \((s^{n+1}, q^n)\).

This dynamic update eliminates the need for a large set of initial \( \mathcal{S}(m) \) states.
4.2. The three-step heuristic

The general idea of the heuristic we propose to obtain a cyclic schedule consists of three steps: initialization, dynamic lookahead, and post-processing.

In the initialization step, the Lower–bound Problem (9)–(10) is solved to obtain $V_1, \ldots, V_J, \theta_1, \ldots, \theta_T$ and $\xi$. Then, the state space is approximated as outlined in Section 3.3. If all setup times are zero, the initial (and final) approximation is given by a set of $W_q = I$ inventory levels for all $q$. Given positive setup times, an initial approximation is computed by using inventory levels visited on one or more arbitrary (given not necessarily optimal) cyclic schedules. (If no good cyclic schedule is known beforehand, the default initial approximation suggested in Section 3.3 can be used.) A random initial state $(s^1, m^1)$ is selected.

In step 2, $K$ steps of the $M$-step Lookahead Problem (17)–(21) are performed to find a sequence of products and associated idling and production times. The states visited in the lookahead heuristic are used dynamically to update the state-space approximation before the system moves to the next state. In other words, given state $(s^1, m^1)$:

1. The $M$-step Lookahead Problem (17)–(21) is solved for a given approximation $\tilde{S}^{W_1}(1), \ldots, \tilde{S}^{W_1}(I)$.
2. The sets $\tilde{S}^{W_1}(1), \ldots, \tilde{S}^{W_1}(I)$ are updated as indicated in Corollary 1: if one of the states $(s^q, q^n)$, $n = 2, \ldots, M$, is outside the current state approximation, $W_q$ is increased by one and its inventory vector $s^q$ is added to $\tilde{S}^{W_q}(q^n)$.
3. Action $(u^1, q^1, t^1)$ is implemented. The state is updated to be $(s^1, m^1) = (s^2, q^1)$.

Steps (1)–(3) are repeated $K$ times and the history of states visited is recorded.

The post-processing step checks whether a state has been visited multiple times. If yes, a cyclic schedule is identified: Its production sequence $q$ is given by the machine states between the two occurrences. If not, let $q$ be the production sequence given by the last $K' \leq K$ machine states visited. Given production sequence $q$, a nonlinear problem is solved to find cost-minimizing idling and production times. (The nonlinear problem is formulated in online Appendix B.)

The reason for the possible discrepancy between the cost generated directly by our heuristic and the “optimized” cost is that the heuristic may converge to non-optimal inventory levels given $q$. Two separate effects can cause this behavior. First, the state-space approximation with a short lookahead (low $M$) can prevent the heuristic from exploring the necessary part of the state-space. Second, given the initial state, the heuristic might act too greedily to incur high immediate costs to reach a better inventory position later on, especially if $M$ is low. The reader is referred to online Appendix D for an example illustrating all three steps of this heuristic.

5. Computational experiments

If there exists a cyclic schedule with non-zero production times and average total cost that is equal to the solution of (9)–(10), one can show that, under a condition on the setup times, the price-directed heuristic will always find an optimal cyclic schedule if it is initialized appropriately, as discussed in the proof of Theorem A1 in online Appendix E. Dobson’s (1992) heuristic will only find this optimal cyclic schedule if it is a power-of-two schedule. In the online appendix, we prove optimality of our heuristic within this class of problems and give such an example, in which Dobson’s heuristic yields arbitrarily poor results. The proof showing optimality of the price-directed heuristic is similar to the one-product case but requires a formal introduction of the semi-infinite linear program (LP) and its dual.

In many realistic problems, however, the lower bound is not tight and we do not have such a guarantee. This is why we tested the proposed algorithm on two variations of five different problems: For each problem, we consider both a version with positive and a version with zero setup times in an attempt to separate the effects of the value function approximation from the effect of the state space approximation. (Remember that a state space approximation is not needed if the setup times are zero.) In particular, we consider: (i) the two-product example given in Boctor (1982), (ii) the three product example discussed in Sing and Foster (1987), (iii) the real company example given in Mallya (1992), (iv) a heavily sequence-dependent five-product example, and (v) the famous Bomberger (1966) problem set.

The data for all problems are given in online Appendix F. For each example, we pick a starting state on the trajectory of the best rotation schedule. We then calculate the lower bound to obtain the values of $\xi$, $V_i$, and $\theta_i$ for all $i \in I$. If setup times are positive, we use the default initial state-space approximation suggested in Section 3.3. $K = 2500$ steps of a two-step lookahead, Equations (17)–(21) with $M = 2$, are used in all examples. The inventory vectors visited by the lookahead heuristic are used to update the state-space approximation dynamically.

Tables 2 and 3 give an overview of the comparative total cost values. We report the lower bound, the average total cost of the best rotation schedule, and the average total cost of Dobson’s static heuristic. In the “lookahead” column we report the average total cost incurred by the dynamic heuristic directly, before the post-processing stage. It states the average total cost of the resulting cyclic schedule in case of convergence or the average total cost over the final $K' = 1000$ steps in case of non-convergence within 2500 steps. Non-convergence is indicated by an asterisk next to the reported number. We report the average total cost after the post-processing step in the last column.

Note that the Three-Step Heuristic may occasionally return lower total cost for a problem with positive setup times than for the same problem with zero setup times. In these
instances, one could simply use the cyclic schedule of the positive setup time case, replace setup times by idling and thus generate an improved schedule for the zero setup time case. In the following, however, we report the result that is obtained directly by following the Three-Step Heuristic.

All costs are presented in units of the lower bound (i.e., average total cost/bound) so this value will be one if the lower bound is achieved and is greater than one otherwise. The lower bound of the positive and zero setup time cases need not be identical, in general. Because most real world examples have relatively low average machine utilization, however, it is unsurprising that this is the case in all five examples.

Either a mixed integer solver like KNITRO can be used to solve Program (17)–(21) for each step or a general nonlinear solver like MINOS can solve for production and idling times given a fixed sequence. The solution time for one step varies greatly depending on the number of products, the number of points used for the state-space approximations, and the solver (options) used, ranging from less than 1 to about 15 seconds per step in the problems described. Obviously, total solution times also heavily depend on the number of steps simulated.

In the following, we discuss each problem in detail. We will often refer to the “average total cost” of a schedule as the “cost” of a schedule for the sake of brevity.

5.1. Boctor’s (1982) problem

In his article, Boctor (1982) suggests a non-trivial example of the sequence-independent ELSP with two products. For his problem, the lower bound is 118.32. The best rotation schedule achieves cost of 119.16, or 1.0071 times that lower bound.

Starting the two-step lookahead, the algorithm quickly converges to a schedule with production sequence \( q = (1, 1, 2, 1, 2, 1, 2) \) and cost of 118.50. As a result of the post-processing step, we obtain a static cyclic schedule with a cost of only 118.38. Even the schedule obtained directly by the lookahead heuristic generates lower average total cost than the best static schedule found by Boctor (1982), which he claims to be optimal at a cost of 118.7. The small difference between the pure lookahead and the optimized costs comes from the difficulty in choosing the optimal balance of idling and production starting from inventories that were optimal for a rotation schedule. If the heuristic were started from a different point, this difference can go to zero. The state-space approximation seems to play no major role in the search.

In the zero setup times case, the lower bound is identical at 118.32 and a rotation cycle yields a cost of 119.16. Starting with a rotation schedule, the dynamic procedure converges to a production sequence of \( q = (1, 1, 2, 1, 2, 1, 2, 1, 2) \) at a cost of 118.49. After the post-processing step, the cyclic schedule based on this sequence gives a slightly lower cost of 118.42.

5.2. Sing and Foster’s (1987) problem

Sing and Foster (1987) discuss a sequence-dependent three-product problem. The lower bound for the average total cost is 21563 with zero, and also with positive, setup times. In its original formulation with positive setup times, the best rotation schedule generates a cost of 24622, Dobson’s heuristic produces a cost of 22543.

In this example, the two-step lookahead heuristic does not have the power to deviate from the initial rotation schedule because it acts too greedily and strictly cycles from \( \tilde{s}_w^w \) to \( \tilde{s}_r^w \). One major factor that makes the state space exploration difficult is that there is no idling in the original production sequence, which limits flexibility to pursue different schedules.

Two fixes are possible to mitigate this inferior behavior. First, one could initialize the state space with more states than only those given by a rotation schedule. This might give the two-step lookahead more flexibility to find better schedules. Second, one could increase \( M \) to look ahead more steps. Experiments showed that a six-step lookahead had the power to extend the state-space sufficiently to reach the same sequence as Dobson’s heuristic starting from a rotation schedule.

If the setup times are zero (i.e., if the state space need not be approximated) the two-step lookahead converges to a schedule with production sequence \( q = (1, 2, 3, 1, 2, 1, 2) \) at a cost of 21576 carrying slightly higher inventory levels than optimal. After the post-processing step, our procedure and Dobson’s heuristic find the same schedule with cost of 21571.

5.3. Mallya’s (1992) problem

Mallya (1992) and Moon (1993) discuss data from a small-to-medium-sized British mechanical engineering production facility. The machine under consideration produces five different products. Both setup times and costs are incurred, but they are sequence-independent. The lower bound for the average total cost is 39.305. The rotation schedule is identical to Dobson’s in this case producing average total cost of 41.759. Before the post-processing step, the two-step lookahead heuristic yields the same production sequence with a somewhat higher cost of 45.747. After the post-processing step, the heuristic yields Dobson’s schedule and the same cost.

If all setup times are set to zero, the bound is still 39.305 and the best rotation and the Dobson schedules are unchanged. Although the values of the \( V_i \), \( \theta_i \), and \( \xi \) are unchanged, our two-step lookahead heuristic now has more freedom to explore different sequences. Remember that we do not force a cyclic behavior of any kind in the lookahead step of our heuristic, so it is not surprising that even
A price-directed heuristic for the ELSP

5.4. A highly sequence dependent problem

Consider the following, highly sequence-dependent, five-product instance, where setups from products 1 and 2 to 4 are as cheap as setups from 3 and 4 to 5. From product 5, setups to products 1, 2, and 3 are cheap and all other setups are expensive. In particular, we choose \( c_{14} = c_{24} = c_{35} = c_{45} = c_{51} = c_{52} = 190, c_{i} = 380 \) for all \( i \in I \) and \( c_{ij} = 570 \) for all other \( i, j \in I \). Setups to product 5 and from product 3 are long; all other setup times are short. We choose \( \tau_{14} = 1, \tau_{5} = 2 \) and \( \tau_{ij} = 0.2 \) for all \( i = 1, 2, 4, 5 \) and \( j = 1, 2, 3, 4 \). Further, we let \( \lambda_{i} = 1, p_{i} = 20 \) for all \( i \in I \), and \( \phi_{1} = \phi_{2} = \phi_{3} = 1, \phi_{4} = 4, \) and \( \phi_{5} = 9 \). The lower bound is 152. It is obvious that a rotation schedule would do poorly in this example, Dobson's cyclic schedule gives a cost of 184.46. A two-step lookahead converges to a schedule with sequence \( q = (5, 2, 4, 5, 3, 5, 1, 4) \) generating average total cost of 153.71. The post-processing step yields a schedule with cost as low as 40.674.

In this example, the setup costs were chosen such that the triangle inequality \( c_{ij} \leq c_{ik} + c_{kj} \) is fulfilled for all \( i, j, k \in I \). If we gave up this requirement and replaced the costs of 380 and 570 by larger numbers, neither the bound nor the cost produced by the dynamic lookahead heuristic would change, but the cost produced by both the rotation cycle and the best Dobson schedule would become larger. For a value of 5000 for those setup costs, both the best rotation schedule and the Dobson schedule would operate at average cost of more than three times the lower bound.

5.5. Bomberger’s (1966) problem

To demonstrate the general applicability of the heuristic to larger problems, we consider the 10-product data first suggested by Bomberger (1966). This sequence-independent problem is probably the most commonly cited data and has a lower bound of 16.87 (with and without setup times). The best rotation schedule yields a cost of 22.50, Dobson's schedule yields 17.18, and other authors have identified schedules with costs as low as 16.90 (e.g., Grznar and Riggle, 1997). Since the overall utilization is low in this example, neither the cost of the rotation nor the cost of the Dobson heuristic is affected by positive setup times.

Although a two-step lookahead seems to be very short-sighted for instances with many products, we keep \( M = 2 \) for the sake of consistency. No cyclic schedule is reached within 2500 iterations (whether setup times are zero or positive). The cost incurred during the final 1000 steps was 17.21. Given the sequence dictated by the last 1000 steps, the post-processing step gives a schedule with a cost of 16.99.

With zero setup times the algorithm produces a schedule with a cost of 17.20. The cost of a cyclic schedule that is

Table 2. Comparison of average total cost in problems with positive setup times

| Problem | Lower bound | Rotation | Dobson | Lookahead | Three-step heuristic |
|---------|-------------|----------|--------|-----------|---------------------|
| 1       | 118.32      | 119.16(1.0071) | 119.16(1.0071) | 118.50(1.0015) | 118.38(1.0005)     |
| 2       | 21.563      | 24.622(1.1419) | 22.543(1.0455) | 24.622(1.1419) | 24.622(1.1419)     |
| 3       | 39.305      | 41.759(1.0625) | 41.759(1.0625) | 45.747(1.1639) | 41.759(1.0625)     |
| 4       | 152.00      | 214.96(1.4142) | 184.46(1.2135) | 153.71(1.0113) | 152.29(1.0019)     |
| 5       | 16.873      | 22.50(1.3338)  | 17.18(1.0183)  | 17.21(1.0199)* | 16.99(1.0068)      |

Table 3. Comparison of average total cost in problems with zero setup times

| Problem | Lower bound | Rotation | Dobson | Lookahead | Three-step heuristic |
|---------|-------------|----------|--------|-----------|---------------------|
| 1       | 118.32      | 119.16(1.0071) | 119.16(1.0071) | 118.49(1.0015) | 118.42(1.0009)     |
| 2       | 21.563      | 22.954(1.0645) | 21.571(1.0004) | 21.576(1.0006) | 21.571(1.0004)     |
| 3       | 39.305      | 41.759(1.0625) | 41.759(1.0625) | 44.088(1.1217)* | 40.674(1.0348)     |
| 4       | 152.00      | 214.96(1.4142) | 165.64(1.0897) | 156.33(1.0285) | 152.00(1.0000)     |
| 5       | 16.873      | 22.50(1.3338)  | 17.18(1.0183)  | 17.20(1.0195)* | 17.05(1.0109)      |
based on the production sequence of the last 1000 steps is 17.05.

6. Outlook

We presented a dynamic, price-directed heuristic for the classic sequence-dependent ELSP. Using a new formulation of a well-known lower bound, we showed how the solution can directly be implemented in a dynamic, price-directed heuristic. To our knowledge, this work is the first to suggest the use of values generated by a lower bound to the sequence-dependent ELSP directly in the construction of a dynamic policy, which is then used to obtain a cyclic schedule.

We can show that there is a class of problems for which the price-directed heuristic can find an optimal cyclic schedule, whereas Dobson’s heuristic (Dobson, 1992) will only find this schedule if the optimal production frequencies are powers of two. Our computational experiments show that the heuristic performs competitively when used for other problems as well, especially when setup times are relatively small. In the 10 instances considered, our heuristic leads to lower total cost than Dobson’s schedule seven times and ties twice. This is particularly noteworthy because Dobson’s heuristic enumerates a couple of alternative production schedules and picks the one with the overall lowest average total cost, whereas our heuristic dynamically constructs and evaluates only one sequence.

Furthermore, one should keep in mind that we used the most basic setup possible to generate our numerical results. We started our algorithm with a state-space approximation with $W_i = 1$ for all products $i \in I$ only. Using $M = 2$ in the lookahead is the smallest value we can use to allow for state-space exploration. The price-directed lookahead heuristic should be expected to prescribe lower total cost, and hence lead to better schedules when $M$ is increased and/or more inventory vectors are used in the initial state-space approximation.

Increasing the number of steps taken by the lookahead should also solve the main issue we identified for this heuristic, namely, that production times are systematically underestimated and, as a consequence, the state-space is not explored sufficiently to find optimal schedules for small values of $M$. A direct implementation of the mixed-integer nonlinear problem given in Problem (17)–(21), however, is NP-hard and computationally expensive for large $M$. More research is needed to improve the efficiency of solving Problem (17)–(21). We leave this direction for future work.

Another promising approach to improve the performance of the price-directed heuristic is to consider a rollout heuristic (e.g., Bertsekas, 2010). Future research also could directly address the issue of underestimated production times when the bound is loose. It seems that the $V_i$ values should be increased, but it is unclear by how much. To improve the state-space approximation without increasing $M$, further research could also address the structure of the state-space.

Concerning the model, the most striking setback of our approximation is that it is linear in the inventory levels. More general approximations of the value function would most likely translate into a tighter bound and an improved dynamic policy.

Finally, we only considered the classic version of the ELSP with static, deterministic demand and no stockouts. A straightforward way to extend the current model to capture backorders is to add the incremental backorder costs of product $q$ incurred by the production decision $(u, q, t)$ to $\kappa(s, m, u, q, t)$. Using the same affine approximation of the bias function, one can derive the sequence-dependent version of the Lower-bound problem developed in Gallego and Roundy (1992). Using this cost accounting in a dynamic policy, however, leads to a defective policy which might never set up for a product that is backordered. More research is needed to investigate if the same lower bound could be derived by different cost accounting and if that would lead to a better policy. A more general approximation of the value function, which uses different values for inventories and backorders, or values that depend on the magnitude of the state, should also be considered.

Funding

Daniel Adelman thanks the University of Chicago Booth School of Business for financial support of this research.

References

Adelman, D. (2004a) A price-directed approach to stochastic inventory/routing. Operations Research, 52, 499–514.
Adelman, D. (2004b) Price-directed replenishment of subsets: methodology and its application to inventory routing. Manufacturing & Service Operations Management, 5, 348–371.
Adelman, D. and Barz, C. (2014) A unifying approximate dynamic programming approach to the economic lot scheduling problem. Mathematics of Operations Research, 39, 374–402.
Adelman, D. and Nemhauser, G.L. (1999) Price-directed control of remnant inventory systems. Operations Research, 47, 889–898.
Altiok, T. and Shiue, G.A. (1994) Single-stage, multi-product production/inventory systems with backorders. IIE Transactions, 26, 52–61.
Anderson, E.J. (1990) Testing feasibility in a lot scheduling problem. Operations Research, 38, 1079–1088.
Aragone, L.S. and Gonzalez, R.L.V. (2000) A fast computational procedure to solve the multi-item single machine lot scheduling optimization problem: the average cost case. Mathematics of Operations Research, 25, 455–475.
Bertsekas, D.P. (2005) Dynamic programming and suboptimal control: a survey from ADP to MPC. European Journal on Control, 11, 310–334.
Bertsekas, D.P. (2007) Dynamic Programming and Optimal Control: Volume 2. Athena Scientific, Nashua, NH.
Bertsekas, D.P. (2010) Rollout algorithms for discrete optimization: a survey, in Handbook of Combinatorial Optimization, Pardalos,
Bertsekas, D.P. and Tsitsiklis, J.N. (1996) Neuro-Dynamic Programming, Athena Scientific, Nashua, NH.

Boctor, F.F. (1982) The two-product, single machine, static demand, infinite horizon scheduling problem. Management Science, 28, 787–807.

Bomberger, E.E. (1966) A dynamic programming approach to a lot size scheduling problem. Management Science, 12, 778–784.

Carstensen, P. (1999) The economic lot scheduling problem—survey and LP-based method. OR Spektrum, 21, 429–460.

Davis, S.G. (1995) An improved algorithm for solving the economic lot size problem (ELSP). International Journal of Production Research, 33, 1007–1026.

Dobson, G. (1992) The cyclic lot scheduling problem with sequence-dependent setups. Operations Research, 40, 736–749.

Eisenstein, D.D. (2005) Recovering cyclic schedules using dynamic produce-up-to policies. Operations Research, 53, 675–688.

Elmaghraby, S.E. (1978) The economic lot scheduling problem (ELSP): review and extensions. Management Science, 24, 587–598.

Faaland, B.H., Schmitt, T. and Arreola-Risa, A. (2004) Economic lot scheduling with lost sales and setup times. IIE Transactions, 36, 629–640.

Gallego, G. (1990a) An extension to the class of easy economic lot scheduling problems. IIE Transactions, 22, 189–190.

Gallego, G. (1990b) Scheduling the production of several items with random demands in a single facility. Management Science, 36, 1579–1592.

Gallego, G. (1994) When is a case stock policy optimal in recovering disrupted cyclic schedules? Naval Research Logistics, 41, 317–333.

Gallego, G. and Roundy, R. (1992) The economic lot scheduling problem with finite backorder costs. Naval Research Logistics, 39, 729–739.

Gallego, G. and Shaw, D.X. (1997) Complexity of the ELSP with general cyclic schedules. IIE Transactions, 29, 109–113.

Gans, N. and van Ryzin, G. (1999) Dynamic vehicle dispatching: optimal heavy traffic performance and practical insights. Operations Research, 47, 675–692.

Goncalves, J.F. and Leachman, R.C. (1998) A hybrid heuristic and linear programming approach to multi-product machine scheduling. European Journal of Operational Research, 110, 548–563.

Grznar, J. and Riggle, C. (1997) An optimal algorithm for the basic period approach to the economic lot scheduling problem. Omega, 25, 335–364.

Gupta, D. (1992) On the economic lot scheduling problem with backlogging: the common cycle approach. Operations Research Letters, 12, 101–109.

Hodgson, T.J, Ge, G. King, R.E., and Said, H. (1997) Dynamic lot size/sequencing policies in a multi-product single-machine system. IIE Transactions, 29, 127–137.

Hsu, W.L. (1983) On the general feasibility test of scheduling lot sizes for several products on one machine. Management Science, 29, 93–105.

Inman, R.R. and Jones, P.C. (1993) Decomposition for scheduling flexible manufacturing systems. Operations Research, 41, 608–617.

Jans, R., and Degraeve, Z. (2007) Meta-heuristics for dynamic lot sizing: a review and comparison of solution approaches. European Journal of Operational Research, 177, 1855–1875.

Jones, P.C. and Inman, R.R. (1989) When is the economic lot scheduling problem easy? IIE Transactions, 21, 11–20.

Mallya, R. (1992) Multi-product scheduling on a single machine: a case study. Omega, 20, 529–534.

Maxwell, W.L. (1964) The scheduling of economic lot sizes. Naval Research Logistics Quarterly, 11, 89–124.

Mehrotra, M., Dawande, M., Gavirneni, S., Demirci, M. and Tayur, S. (2011) Production planning with patterns: a problem from processed food manufacturing. Operations Research, 59, 267–282.

Monkman, S.K., Morrice, D.J. and Bard, J.F. (2008) A production scheduling heuristic for an electronics manufacturer with sequence-dependent setup costs. European Journal of Operational Research, 187, 1100–1114.

Moon, I. (1993) A note on correction on “Multi-product scheduling on a single machine.” Omega, 21, 597–599.

Powell, W.B. (2007) Approximate Dynamic Programming, John Wiley & Sons, Inc, Hoboken, NJ.

Qui, J. and Loulou, R. (1995) Multiproduct production/inventory control under random demands. IEEE Transactions on Automatic Control, 40, 350–356.

Robinson, L.W., and Chen, L.G. (2008) Quick recovery to alternative ELSP schedules. Working Paper, Johnson School, Cornell University, Ithaca, NY.

Rogers, J. (1958) A computational approach to the economic lot scheduling problem. Management Science, 4, 264–291.

Roundy, R. (1989) Rounding off to powers of two in continuous relaxations of capacitated lot sizing problems. Management Science, 35, 1433–1442.

Schäl, M. (1992) On the second optimality equation for semi-markov decision models. Mathematics of Operations Research, 17, 470–486.

Segerstedt, A. (1999) Lot sizes in a capacity constrained facility with available initial inventories. International Journal of Production Economics, 59, 469–475.

Sing, H. and Foster, J.B. (1987) Production scheduling with sequence-dependent setup costs. IIE Transactions, 1, 43–49.

Soy, C.R., Jackson, P.L., Bowman, A. and Muckstadt, J.A. (1999) A review of the stochastic lot scheduling problem. International Journal of Production Economics, 62, 181–200.

Talluri, K. and van Ryzin, G. (1998) An analysis of bid-price controls for network revenue management. Management Science, Vol.44, 1577–1593.

Taylor, G.D., Taha, H.A. and Chowning, K.M. (1997) A heuristic model for the sequence-dependent lot scheduling problem. Production Planning & Control, 8, 213–225.

Wagner, B.J. and Davis, D.J. (2002) A search heuristic for the sequence-dependent economic lot scheduling problem. European Journal of Operational Research, 141, 133–146.

Winands, E.M.M., Adan, I.J.B.F. and van Houtum, G.J. (2005) The stochastic economic lot scheduling problem: a survey. European Journal of Operational Research, 210, 1–9.

Zhu, X. and Wilhelm, W.E. (2006) Scheduling and lot sizing with sequence-dependent setup: a literature review. IIE Transactions, 38, 987–1007.

Zipkin, P.H. (2000) Foundations of Inventory Management, McGraw-Hill, Singapore.

Biographies

Dan Adelman is a Professor of Operations Management at the University of Chicago Booth School of Business. He has published works on production/distribution/inventory control and revenue management/dynamic pricing motivated by various industry contexts. Throughout much of his work, he advances approximate dynamic programming theory.
methods, and applications. He has held and currently holds several editorial positions in the field.

Christiane Barz is an Assistant Professor of the Decisions, Operations & Technology Management group at the UCLA Anderson School of Management. She holds a doctorate degree and a diploma in Industrial Engineering from the Karlsruhe Institute of Technology, Germany. Her main research interest is approximate dynamic programming with applications in revenue management, healthcare services, and production scheduling.