Spare parts inventory control for an aircraft component repair shop

Willem van Jaarsveld and Twan Dollevoet

Erasmus School of Economics, Erasmus University Rotterdam

Working Paper, Econometric Institute Report EI 2011-24

July 11, 2011

Abstract

We study spare parts inventory control for a repair shop for aircraft components. Defect components that are removed from the aircraft are sent to such a shop for repair. Only after inspection of the component, it becomes clear which specific spare parts are needed to repair it, and in what quantity they are needed. Market requirements on shop performance are reflected in fill rate requirements on the turn around times of the repairs for each component type. The inventory for spare parts is controlled by independent min-max policies. Because parts may be used in the repair of different component types, the resulting optimization problem has a combinatorial nature. Practical instances may consist of 500 component types and 4000 parts, and thus pose a significant computational challenge. We propose a solution algorithm based on column generation. We study the pricing problem, and develop a method that is very efficient in (repeatedly) solving this pricing problem. With this method, it becomes feasible to solve practical instances of the problem in minutes.

Keywords: Continuous Review Assemble-to-Order Systems, Min-Max Policies, Spare Parts, Column Generation

1 Introduction

During line and heavy maintenance, components are removed from the aircraft and repaired separately. The repair of these components generated an annual turnover of $9 billion in recent years, of which 70% is outsourced to (semi-) independent repair shops (AFM 2009). In order to enable the efficient planning and execution of aircraft maintenance, operators use their bargaining power to pressure repair shops to attain short and reliable repair turn around times (TATs) for the components. In case of in-house shops, the need for efficient line maintenance planning is typically more directly reflected in business targets on repair TATs.

The most challenging aspect of guaranteeing reliable repair TATs is assuring the timely availability of the spare parts needed in the repairs. Only after inspection of the component at the repair shop, it becomes clear which specific spare parts are needed to repair it. Spare parts generally have supply lead-times that exceed the time that operators are willing to wait for repairs to finish. To fulfill their customers' needs, repair shops thus need to keep a local stock of spare parts. The inventory must be sufficient to meet the target TATs, while excess inventory may significantly reduce

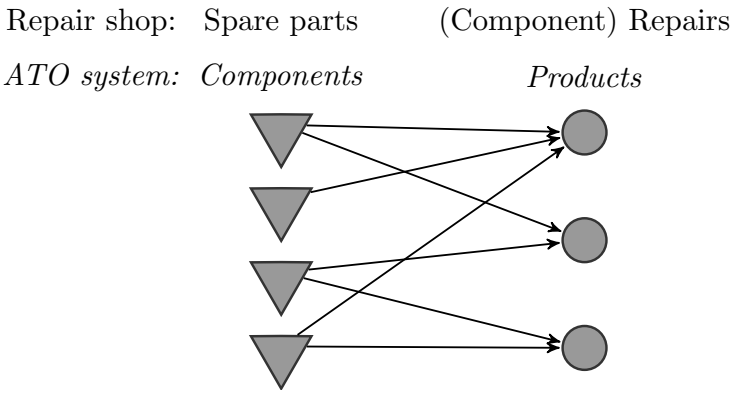

Figure 1: A schematic representation of the repair shop/*ATO system*. Terminology for the repair shop is shown in regular font, while standard ATO terminology is shown in italic.

profits. Therefore, efficient inventory control is an essential requirement in order to competitively perform component repairs.

Components may consist of hundreds of parts, each of which may need replacement to complete a repair. Since a repair shop typically repairs a range of component types, thousands of spare parts need to be stocked in order to guarantee timely repairs. The difficulty of managing such a large assortment is further complicated because parts may be used in the repair of different component types, each of which may have different availability targets. As a result, it is challenging and time-consuming for human decision makers to control inventory in this environment. We develop a model-based method to determine suitable inventory policies for each part. We believe that the use of this method can help in overcoming the above-mentioned challenges of stocking for a repair shop, and as such our research contributes to practice.

From a modeling perspective, the system we consider can be regarded as an Assemble-to-Order (ATO) system. In ATO systems products are assembled from multiple components on demand, while in our setting component repairs require multiple spare parts to complete (see Figure 1). A difference between the system we consider and many other ATO systems lies in the spare parts (components) required to repair a component (assemble a product) of a given type. To repair a component, a large number of spare parts is possibly needed with low probability. To assemble a product, a fixed set of components are typically always needed. Our results are valid for general ATO systems.

A motivation of the study of ATO systems is given by Song and Zipkin (2003). They find that "many real ATO systems contain hundreds of components and thousands of products". The system that we consider is another example of such large-scale systems. In the next section, we will review the literature on ATO systems. Our method appears to be the first method capable to handle problems of this size, at least in the discrete inventory setting.

Under the continuous inventory approximation, some contributions exploit the continuity of the policy space to develop algorithms that might be scalable to handle larger systems, but no tests of the applicability of these algorithms on such systems have been reported (to the best of our knowledge). More importantly, the continuous inventory approximation is unsuitable when the discrete nature of inventory cannot be ignored, which is often the case in practice. E.g. in the setting that we consider, a significant fraction of spare parts (which play the role of components, see Figure 1) is slow moving. Suitable reorder points for these parts are in the $0-4$ range, a continuous approximation is thus clearly inappropriate.

Most studies of ATO systems assume that the inventory of each part is controlled independently; our focus will also be on this type of policies. The study of systems controlled in this manner is very relevant for practice, since independent control policies are used by many companies, mainly because they are easy to implement.

A special type of independent control policies is formed by the independent base-stock policies, resulting in a base-stock ATO system. In the literature review we find that the majority of studies on ATO systems focuses on such policies. The practical value of this approach is limited in many environments, as this approach does not consider the issue of order sizes. This is generally an important issue, as the fixed ordering costs are significant in comparison to the holding costs for at least some (inexpensive) components (spare parts). The problem we consider serves as an example. (Even though the base-stock approach ignores the interaction between safety stock and order sizes, it can be suitable when the focus is solely on gaining insights in the high-level inventory-service trade-off (Song and Yao 2002).) To take into account order sizes, a commonly used approach in practice is to use (s, S) policies. For a discussion of such policies see e.g. Axsäter (2006, pp. 48-49). We take this more general approach (the (R, nQ) case is also shortly discussed). To our best knowledge, we are the first to consider optimization of (s, S) (or (R, nQ)) policies in ATO systems.

Our approach is based on column generation. We use bounds on the fill rate to obtain a surrogate optimization problem, an approach also taken by Zhang (1997), Song and Yao (2002), Cheng, Ettl, Lin and Yao (2002) and Lu, Song and Yao (2005). As a consequence of using bounds, the pricing problem is separable: it reduces to the separate optimization of the inventory parameters for each part. We derive a special-purpose algorithm to solve these subproblems, which is very efficient at repeatedly solving the same problem with a slightly changed cost function. The algorithm works under more general cost functions than the algorithm proposed by Zheng and Federgruen (1991). This latter algorithm is not applicable in our case.

In summary, our contributions is the development of an algorithm for the optimization of inventory control for ATO systems. Unlike previous contributions, the algorithm is capable of handling the size of many real systems. The method is the first to consider optimization of (s, S) policies in an ATO system, which is a significant improvement on base-stock policy optimization in terms of applicability. Additionally, the algorithm that solves the pricing problem in our column generation approach is interesting in its own right, because it works under more general assumptions than existing algorithms.

The rest of this paper is organized as follows. In the next section we review relevant literature. In Section 3 we describe the model, and derive bounds on performance measures. We give a formulation of the optimization problem, and investigate the pricing problem. In Section 4 we describe the algorithm. In Section 5 we test the algorithm on a practical problem, and discuss the performance of the algorithm.

2 Literature review

The optimization and evaluation of ATO systems is generally performed under heuristic policy types, as the structure of the optimal policy is only known in special cases (see e.g. Rosling (1989) for periodic review, deterministic lead-time, single product systems, and Benjaafar and ElHafsi (2006) for continuous review, $M/M/1$ resupply, single product systems). Most studies focus on independent base-stock ATO systems. These studies can be characterized into single period, periodic review and continuous review systems. Since single period systems are not appropriate for our case, we refer to Song and Zipkin (2003, Section 2) for an overview.

We now review studies on periodic review systems. The reviewed studies all assume determin-

istic lead-times and a first come first serve (FCFS) component allocation policy. For allocation to products arriving in the same period, the assumed allocation policy differs for different studies. Evaluation is done under the assumption of multivariate normal demand (with the exception of Akçay and Xu (2004)).

Hausman, Lee and Zhang (1998) focus on the probability that all demand in a period is filled within a given time window. For cost minimization with a constraint on this performance measure, a heuristic is proposed which uses an equal fill rate for each component. The approach is limited in its use because it cannot properly account for different fill rate targets for different products.

Zhang (1997) assumes fixed priority allocation, and considers cost minimization with product-specific fill rate restrictions. It is shown that the feasible region for the problem is convex, and an optimal solution is determined by employing a feasible direction algorithm. Agrawal and Cohen (2001) find similar results under a fair share allocation rule.

Cheng et al. (2002) study a PC assembly system. As Zhang (1997); they study the product-specific fill rate constrained cost minimization problem. Special purpose algorithms are developed, based on a lower bound on the fill rate. One proposed algorithm is only applicable under the restrictive assumption that each product uses a unique component, in which case it can be shown that all constraints are binding. The algorithm resembles sub-gradient methods, in the sense that it makes incremental updates of the Lagrangian multipliers. This algorithm is tested for a 18 product, 17 component system, but no computation times are reported so it is unclear whether the algorithm scales to larger instances. For the general case, a greedy heuristic is proposed, which remains untested.

Akçay and Xu (2004) consider time-window based, weighted fill rate maximization under a budget constraint. Unlike the studies discussed earlier, the allocation rule in their approach is dynamic. Moreover, analysis is not restricted to a specific demand distribution. The problem is modeled as a two-stage stochastic program. A sample average approximation is employed to find a solution. Unfortunately, this algorithm is not scalable to larger instances, because the number of scenarios required to decently represent the possible behavior grows rapidly for larger systems. Solving the integer problem associated with a sample quickly becomes intractable when the number of scenarios grows.

We now review base-stock ATO systems in the continuous review setting. Reviewed studies assume Poisson demand for finished products. For the assembly of a product of a given type, a predefined (sometimes generalized to random) integer number of components of various types is needed. Finally, components are allocated on a FCFS basis.

Song and Yao (2002) consider a single product system under independent identically distributed (iid) component lead-times. They consider budget-constrained back-order minimization, where they use bounds on the number of back-orders as surrogate objective functions. Algorithms are proposed to solve this problem. Along the same lines, they propose algorithms for inventory minimization under a (surrogate) fill rate constraint.

The multi-product extension is studied by Lu et al. (2005). They consider budget constrained backorder minimization, where again bounds on the expected number of back-orders are used as a surrogate objective function. The resulting problem has a so-called stack structure. As a result, the problem can be solved by solving $k!$ greedily solvable subproblems, where k denotes the number of products. Lu and Song (2005) consider order-based cost minimization for the same system, under the restrictive assumption that each product uses either 1, or 0 of each component (a 0-1 BOM (bill of material)). Backorder costs are paid per product back-ordered per time unit. They derive various properties of the cost function, based on which an optimization approach is formulated. To optimize, the algorithm evaluates the costs of $m^7 \log m$ solutions, where m is the number of components.

Gallien and Wein (2001) consider a single product system. A different policy class is considered: product base stock policies with component order postponement. They use an order synchronization assumption to find an explicit expression for a policy that is near-optimal among policies in this class.

Güllü and Köksalan (2009) consider a system similar to ATO systems, but with a different resupply system. Components that are withdrawn together are replenished together (except for one component in each demand, which is replenished via a separate channel). Exact expressions are derived for the performance of the system. Evaluation of these expressions is not tractable for large scale systems. They propose a greedy type of heuristic for optimization of the base-stock levels.

We conclude that existing algorithms for the optimization in the continuous review setting either are only applicable for single-product systems, or can only be used for relatively small instances.

Other studies in the continuous review setting mainly consider the evaluation of key performance measures such as fill rates and average back-orders in base-stock ATO systems. As exact evaluation is generally intractable for large systems, many contributions also derive more easily computable bounds on and approximations of performance characteristics. In the following, we review such contributions.

For deterministic lead-time systems, Song (1998) focuses on the fill rate and Song (2002) studies the average number of product back orders. Lu, Song and Yao (2003) extend Song (1998) to iid lead-times. Cheung and Hausman (1995) show how to evaluate the average number of customer back orders for a system with iid lead-times, under the assumption of complete cannibalization.

In the make-to-stock setting, Glasserman and Wang (1998) show that there is a linear relationship between delivery time and inventory, in the limit of high fill rates. In the same setting, Song, Xu and Liu (1999) develop methods for exact fill rate evaluation, and Dayanik, Song and Xu (2003) compare different bounds on the fill rate.

Zhao and Simchi-Levi (2006) assume stochastic-sequential lead-times, with a restriction to the 0-1 BOM case. They evaluate expected fill rate and back-orders, and develop approximations. Zhao (2009) extends these results to the general BOM case.

The above studies, with the exception of Zhao and Simchi-Levi (2006) and Zhao (2009), all ignore the issue of batch sizes and focus exclusively on base-stock ATO systems. Song (2000) considers (R, nQ) systems, and finds that their analysis reduces to the analysis of base-stock ATO systems under general conditions. But, as Zhao and Simchi-Levi (2006) point out, the evaluation of a single (R, nQ) policy in this way requires the evaluation of a number of base-stock ATO systems that is exponential in the number of components. To cope with this difficulty, they therefore propose efficient (Monte-Carlo) sampling procedures for evaluation of batching policies.

Our approach is based on column generation. To the best of our knowledge, this study is the first to propose such an approach in an ATO setting. Kranenburg and van Houtum (2008) use column generation for inventory optimization in a multi-item system with multiple demand classes. Kranenburg and van Houtum (2009) use such an approach for optimization of base-stock policies in a multi-echelon inventory system with partial pooling.

3 The model

We consider a repair shop at which components of various types are repaired. Components of different types needing repair arrive according to a Poisson process with known rate. Upon arrival of a repair, inspection reveals which spare parts are needed to repair the component.

Spare parts are stocked in a local warehouse. To control inventory, independent (s, S) policies

are used. Under an (s, S) policy, when the inventory position (= inventory on hand + inventory on order − backlogs) is at or below s, an order is placed to raise it to S. Because delaying the placement of a replenishment order for a part with backlogs is uncommon in practice, we assume $s \geq -1$ (the algorithm extends to more general lower bounds on s with some additional effort).

We assume stochastic sequential lead-times. Svoronos and Zipkin (1991) give a precise definition of such lead times, and argue that this may be a more realistic assumption than iid lead times. We assume the supplier delivers the orders in full. The lead time distribution may be different for different parts. We make no restrictive assumptions on the form of the lead time distribution.

Spare parts are allocated on a FCFS basis to repairs. This allocation policy is commonly used in ATO practice and literature (for exceptions see Lu, Song and Zhao (2010), and references therein). When some parts for a repair are available but others are not, the available parts are put aside as committed inventory (the same assumption is made by e.g. Song (2002) and Zhao and Simchi-Levi (2006)).

We denote the set of spare parts by $\mathscr{J}$. The component repair types are denoted by $\mathscr{I}$. We define the following notation:

- $h_j > 0$: inventory holding costs per unit of time per unit of inventory of part $j \in \mathscr{J}$.
- $o_j \geq 0$: the fixed ordering costs for placing an order for parts of type $j \in \mathscr{J}$.
- $\mathscr{C}_j$: the set of policies for part $j \in \mathscr{J}$. For each valid combination of s and S for part j, we have $(s, S) = c \in \mathscr{C}_j$.
- $t^i(n)$: (random) time of arrival of the nth repair of type $i \in \mathscr{I}$.
- $\mathscr{I}_j$: set of repair types in which parts of type j *may* be used. (We allow $\mathscr{I}_j = \mathscr{I}$, but in practice, parts are only used in a limited range of repair types.)
- $\mathscr{J}^i$: set of parts that may be used in a repair of type i.
- $Y^i(n) = (Y^i_j(n), j \in \mathscr{J}^i)$: random variable indicating the spare parts needed in the nth repair of type $i \in \mathscr{I}$. Here, $Y^i_j(n) (\in \{0, 1, 2, \ldots\})$ denotes the number of parts of type j needed in the nth repair of type i. We assume that $Y^i(n)$ for $n \in \{1, 2, \ldots\}$ are iid random variables (we allow for dependence between $Y^i_j(n)$ for different parts j).
- λ^i: the Poisson arrival rate of repairs of components of type i.
- $\lambda_j = \sum_{i \in \mathscr{I}_j} \lambda^i \mathbf{P}(Y^i_j > 0)$: the rate at which repairs arrive that require a *positive* number of parts of type j. Also: the demand rate for part j.
- $I(t) = (I_j(t, c_j), j \in \mathscr{J})$: (random) inventory on hand vector at time t. The dependence of I_j on the policies $c_j \in \mathscr{C}_j$ will be dropped where no confusion can arise.
- $P(t) = (P_j(t, c_j), j \in \mathscr{J})$: (random) number of purchase orders placed for part j in the time period $(0, t)$.
- $W^i(n)$: (random) waiting time until all spare parts needed in the nth repair of a component of type i are available. For an arbitrary repair as $n \to \infty$, we drop the dependence on n.

The total TAT of the repair involves other activities besides waiting for spare parts availability (such as transport, and actually replacing the parts), but the time required for these activities is generally short and has low variability. As a consequence, the performance targets on the total TAT are easily translated to targets on the waiting time distribution.

3.1 Bounds on performance measures

Repairs of a given type typically may require a broad range (> 50) of spare parts, each with low probability. As a result of the dependence between the inventory level of different parts, exact evaluation of the time-window based fill rate $\mathbf{P}(W^i < w)$ or expected waiting time $\mathbf{E}(W^i)$ for such repair types is intractable (see e.g. Song (1998) and Song (2002)). A well-established method to cope with this difficulty is the use of bounds on the performance measures (see Section 2). We will now derive bounds on the performance of (s, S) ATO systems.

We first derive a bound on the fill rate. We concentrate on bounds on the immediate fill rate, because the fill rate within a time window corresponds to the immediate fill rate in a system with revised lead times. (For details see Proposition 1.1 of Song (1998), which extends with little difficulty to stochastic sequential lead times.)

For the nth repair of type i, $\mathbf{P}(I_j(t^i(n)) < Y_j^i(n))$ equals the probability that the waiting time for parts of type j is positive. We thus have

$$\begin{aligned}\mathbf{P}(W^i(n) = 0) &= 1 - \mathbf{P}\left(\bigcup_{j \in \mathscr{J}^i} I_j(t^i(n)) < Y_j^i(n)\right) \\ &\geq 1 - \sum_{j \in \mathscr{J}^i} \mathbf{P}(I_j(t^i(n)) < Y_j^i(n)), \end{aligned} \tag{1}$$

where the inequality is typically referred to as Boole's inequality. By taking the limit $n \to \infty$, we obtain a bound on the long-term fill rate. Note that this bound is tight if at most a single part causes waiting time in each repair, which may be a good approximation at the optimal solution for reasonably high fill rates.

For the waiting time, we have:

$$\mathbf{E}(W^i) = \int_{w=0}^{\infty} (1 - \mathbf{P}(W^i \leq w))dw \tag{2}$$

$$\leq \sum_{m=2}^{M} (w_m - w_{m-1})(1 - \mathbf{P}(W^i \leq w_{m-1})), \tag{3}$$

for increasing $w_1, w_2, \ldots, w_M$, with $w_1 = 0$ and $\mathbf{P}(W^i \leq w_M) = 1$. The bound in (1) can subsequently be used in the summand, to obtain an efficiently calculable lower bound on the average waiting time. ((2) is used by Song (2002) as a starting point for exact analysis of the expected waiting time.)

We now make a brief excursion to discuss the case of (R, nQ) policies, which are also common in practice. For the non-unit demand case that we consider, such policies are different from (s, S) policies (see e.g. Axsäter (2006, pp. 48-49)).

While the bound (1) remains valid for (R, nQ) policies, this bound can be strengthened when the inventory position has uniform equilibrium distribution (see Song (2000) for conditions). If in addition $Y_j^i(n), j \in \mathscr{J}^i$ are associated (e.g. independent), then $Y_j^i(n) - I_j(t^i(n)), j \in \mathscr{J}^i$ are also associated, in the limit $n \to \infty$. The proof is along the lines of the proof of Proposition 5.1 of Song (1998), we omit details. As a result, the following bound holds:

$$\mathbf{P}(W^i = 0) \geq \prod_{j \in \mathscr{J}^i} \lim_{n \to \infty} \mathbf{P}(I_j(t^i(n)) \geq Y_j^i(n)). \tag{4}$$

A surrogate constraint based on this bound can be linearized by taking the logarithm on both sides (cf. Song and Yao (2002)). Note that the bound in (4) is *not* generally valid for (s, S) policies (we have found an example where it is not valid).

The algorithm developed in Section 4 can be applied for (R, nQ) policies with only limited adaptations. Let $\mathscr{C}_j$ be the set of (R, nQ) policies for part j, and use the correspondence $R \leftrightarrow s, R + Q \leftrightarrow S$ when solving the pricing problem.

3.2 Cost minimization under fill-rate constraints

The focus of the remainder of this paper will be on cost minimization under repair type specific fill rate constraints. The approach can be extended to include constraints on the average waiting time, on the time-window based fill rate, or combinations of such constraints. The focus on the immediate fill rate is thus mainly for simplicity of notation and exposition. In addition, the fill rate (probability that all parts for a repair are immediately available) is a performance measure that is easily communicated with managers and customers. The formulation on which we focus is thus easily applicable in practice.

A natural formulation of the problem would use the policies $(s, S) = c \in \mathscr{C}_j$ directly as decision variables. However, such a formulation would be non-linear. Instead, we let $x_{jc} = 1$ indicate that policy $c = (s, S)$ is used for part j, while $x_{jc} = 0$ indicates that policy c is *not* used for part j. The resulting formulation is linear, but we pay the price of an infinite number of decision variables:

$$\min \sum_{j \in \mathscr{J}} \sum_{c \in \mathscr{C}_j} x_{jc}(H_j(c) + O_j(c)), \tag{5}$$

$$\text{s.t.} \sum_{j \in \mathscr{J}^i} \sum_{c \in \mathscr{C}_j} x_{jc} F_j^i(c) \leq 1 - a^i, \qquad i \in \mathscr{I}, \tag{6}$$

$$\sum_{c \in \mathscr{C}_j} x_{jc} = 1, \qquad j \in \mathscr{J}, \tag{7}$$

$$x_{jc} \in \{0, 1\}, \qquad j \in \mathscr{J}, c \in \mathscr{C}_j. \tag{8}$$

a^i represents the target fill rate for repairs of type i, and

$$H_j(c) = H_j(s, S) = \lim_{t \to \infty} h_j \mathbf{E}(I_j(t, c)), \tag{9}$$

$$F_j^i(c) = F_j^i(s, S) = \lim_{n \to \infty} \mathbf{P}(I_j(t^i(n), c) < Y_j^i(n)), \tag{10}$$

$$O_j(c) = O_j(S - s) = \lim_{t \to \infty} o_j \mathbf{E}(P_j(t, c)/t). \tag{11}$$

The formulation uses the bound on the fillrate (1) in a similar fashion as e.g. Zhang (1997), Song and Yao (2002) and Cheng et al. (2002).

Note that (9-11) for given j are independent of the specific policy used for other parts $j' \neq j$.

3.3 The pricing problem

The first step in the algorithm is solving the continuous relaxation of (5-8). This relaxation is obtained by replacing (8) by

$$0 \leq x_{jc} \leq 1 \qquad j \in \mathscr{J}, c \in \mathscr{C}_j. \tag{12}$$

The reduced cost associated with decision variable x_{jc} is given by

$$R_j(c) = R_j(s,S) = H_j(s,S) + O_j(s,S) + \mu_j + \sum_{i \in \mathscr{I}_j} \nu^i F_j^i(s,S), \tag{13}$$

where $\nu^i \geq 0$ is the dual multiplier associated with constraint i in (6), and μ_j is the dual multiplier associated with constraint j in (7).

In this section, we will investigate the problem of finding the column x_{jc} with the lowest reduced costs for given dual multipliers. We show that this problem is equivalent to cost minimization in a single-item inventory problem. We discuss the applicability of existing algorithms for solving this single-item problem. Finally, we discuss some results that form the basis of the algorithm we propose in Section 4.1 for solving this problem.

We first discuss the evaluation of (9-11). Denote by m_k for $k \in \{0, \ldots, S-s-1\}$ the probability to visit inventory position $S-k$ during an arbitrary order cycle. Note that this probability is independent of S. It is easy to recursively evaluate m_k using the compounding distribution of demand for part j (see e.g. Axsäter (2006, pp. 107-109)). This compounding distribution is the mixture of the compounding distribution of the demand for part j resulting from repairs of type i, for all $i \in \mathscr{I}_j$. The expected length of an order cycle is given by M_{S-s}/λ_j, with $M_{S-s} = \sum_{k=0}^{S-s-1} m_k$. The holding costs for general policies can be expressed in terms of the holding costs for $(S-1,S)$ (base-stock) policies as follows:

$$H_j(s,S) = \sum_{k=0}^{S-s-1} m_k H_k(S-k-1, S-k)/M_{S-s}. \tag{14}$$

The same expression holds with F_j^i replacing H_j. (For a discussion of the evaluation of H_j and F_j^i for $(S-1,S)$ policies see the proof of Lemma 1.) Since by definition a single order is placed in each order cycle, we have $O_j(S-s) = o_j\lambda_j/M_{S-s}$.

To show the equivalence of (13) to a single item inventory policy, we rewrite it as

$$R_j(s,S) = O_j(S-s) + \mu_j + \sum_{k=0}^{S-s-1} m_k G(S-k)/M_{S-s} \tag{15}$$

where

$$G(y) = H_j(y-1,y) + \sum_{i \in \mathscr{I}_j} \nu^i F_j^i(y-1,y). \tag{16}$$

The formulation used by others in the study of optimization of (s,S) policies for single-item inventory problems uses $G(y)$ as the starting point of analysis (note that μ_j is constant and can be excluded). We discuss here in detail the study by Zheng and Federgruen (1991), for a discussion of related literature we refer to their literature review. A crucial assumption in Zheng and Federgruen (1991) is that $-G(y)$ is unimodal (this assumption is common in the study of optimization of (s,S) policies). However, $-G(y)$ as given by (16) need not be unimodal. Consequently the algorithm proposed by Zheng and Federgruen (1991) can not guarantee optimality for the problem we consider. This is a known effect of concentrating on the fill-rate as a performance measure. Alternatively, one could focus on the average waiting time for repairs, which could indeed qualify as a reasonable performance measure for our system. While experience from single item systems might lead one to conclude that this would make $-G(y)$ unimodal, we point out such a conclusion is *not* warranted (single spare part, single repair type counter examples exist).

In the next section we will propose a single-item policy optimization algorithm that *does guarantee optimality* even if $G(y)$ is not unimodal. We conclude this section with summarizing some results that form the basis of this method.

Lemma 1. *1. $H_j(S-1,S)$ is increasing and convex in S.*

2. $F_j^i(S-1,S)$ is decreasing in S.

Proof. For fixed lead time, evaluation of $\lim_{t\to\infty} I_j(t)$ is standard. The results then follow from (9) and (10). For the stochastic sequential lead-time case, $H_j(S-1,S)$ and $F_j^i(S-1,S)$ are evaluated by conditioning on the lead time and using the approach for fixed lead time (cf. Zipkin (1986)). The asserted properties are preserved while conditioning. (Note that convexity does not hold for $F_j^i(S-1,S)$, which is the reason that $G(y)$ may fail to be unimodal.) □

Lemma 2. *1. $H_j(s+\Delta,S)$ (respectively $F_j^i(s-\Delta,S)$) is nondecreasing in Δ.*

2. $H_j(s+\Delta,S+\Delta)$ (respectively $F_j^i(s-\Delta,S-\Delta)$) is nondecreasing in Δ.

Proof. The results are a consequence of (14) (see also (19)) and the monotonicity of $H_j(y-1,y)$ and $F_j^i(y-1,y)$ from Lemma 1. (The result is similar to Lemma 0 of Zheng and Federgruen (1991).) □

Lemma 3. *$O_j(S-s)$ is nonincreasing in $S-s$.*

Proof. Immediate from $O_j(S-s) = o_j\lambda_j/M_{S-s}$ and the definition of M_{S-s}. □

Lemma 4. *For every policy (s,S)*

$$H_j(s,S) \geq H_j\left(\frac{s+1+S}{2}-1, \frac{s+1+S}{2}\right), \tag{17}$$

where $H_j(y-1/2, y+1/2)$ for integer y is defined as the average of $H_j(y-1,y)$ and $H_j(y,y+1)$.

For the proof we refer to the appendix.

4 The algorithm

The algorithm to solve (5-8) consists of two steps: we first solve the continuous relaxation (5-7,12) to obtain a lower bound, and then apply a rounding heuristic. To solve the continuous relaxation, we use a column generation approach. We first describe this column generation approach. Then we describe the algorithm to solve the pricing problem. Finally, we describe the rounding heuristic.

Our column generation approach to solve the continuous relaxation can be summarized as follows:

Algorithm 1.

Step 1 *Determine an initial* finite *set of policies $\mathscr{C}_j' \subset \mathscr{C}_j$ for each part j, by executing the initialization step of Algorithm 2.*

Step 2 *Solve the restricted master problem to optimality. I.e. solve (5-7,12), with $\mathscr{C}_j$ replaced by $\mathscr{C}_j'$. For this purpose we use the simplex method (with specific alterations to improve efficiency, that will be omitted for brevity).*

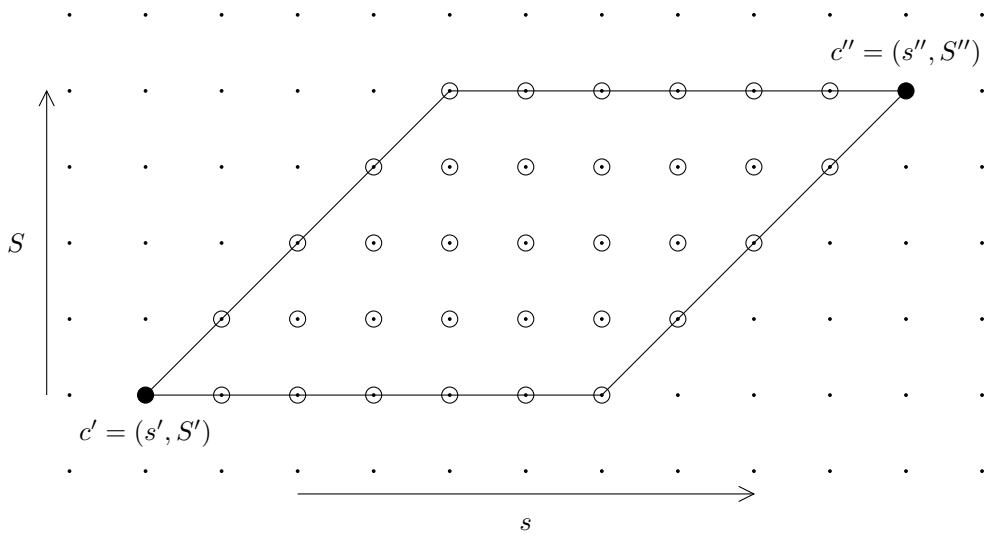

Figure 2: The parallelogram spanned by the policies c' and c''. Policies covered by the parallelogram are circled.

Step 3 *For the dual multipliers ν^i and μ_j obtained in Step 2, execute for each part j Step 1-3 of Algorithm 2. This adds the policy with the lowest reduced cost to $\mathscr{C}'_j$, typically along with other policies that also have low reduced costs.*

Step 4 *If any policies with* negative *reduced costs were added to $\mathscr{C}'_j$ for any part j in the previous step, go to Step 2. Otherwise, terminate.*

When this algorithm terminates, we obtain the optimal solution to (5-7,12). The solution value is a lower bound for the solution value of the integer problem (5-8).

4.1 Algorithm for the pricing problem

To solve the pricing problem in Algorithm (1), we develop an algorithm to determine for a given part $j \in \mathscr{J}$ the (s, S) policy that minimizes the reduced costs, as given by (13), for given dual multipliers ν^i and μ_j. Throughout this section, we will suppress subscript j (in particular, $\mathscr{I}$ will denote $\mathscr{I}_j$).

The novelty of our approach is that it exploits that the cost rate $G(y)$ as a function of the inventory position y is generally made up of two separate constituents, viz. the holding costs and the costs associated with lack of service (see (16)). By Lemma 1, certain properties hold for each of these constituents. (As long as the total cost rate $G(y)$ for being in state y is made up two separate constituents for which the results of Lemma 1 hold, the algorithm applies.)

In particular, the algorithm is based upon the following key observation.

Lemma 5. *Let two policies $c' = (s', S')$ and $c'' = (s'', S'')$ with $S' - s' \geq S'' - s''$ and $S'' \geq S'$ be given. Consider all policies covered by the parallelogram spanned by the policies (s', S') and (s'', S'') in the s, S plane (circled in Figure 2). The reduced costs for these policies is bounded below by*

$$\underline{R}(c', c'') = H(s', S') + O(S' - s') + \sum_{i \in \mathscr{I}} \nu^i F^i(s'', S'') + \mu \tag{18}$$

Proof. Let (s, S) denote a policy. By Lemmas 2 and 3,

1. Suppose there exist $\Delta, \Delta' \geq 0$ such that $(s, S) = (s' + \Delta' + \Delta, S' + \Delta')$. Then $H(s, S) \geq H(s', S')$;
2. Suppose there exist $\Delta, \Delta' \geq 0$ such that $(s, S) = (s'' - \Delta' - \Delta, S'' - \Delta')$. Then $F^i(s, S) \geq F^i(s'', S'')$;
3. Suppose there exist $\Delta \leq S' - s'$ and s such that $(s, S) = (s, s + \Delta)$. Then $O(s, S) \geq O(s', S')$.

All policies enclosed in the parallelogram satisfy Hypotheses 1, 2 and 3. Consequently, their reduced costs are bounded below by (18) (note that $\nu^i \geq 0$). □

The algorithm we propose is based on a grid of parallelograms, each of the type described in Lemma 5. We denote such a parallelogram by $g = (c'(g), c''(g)) = (s', S'; s'', S'')$, with $c'(g)$ and $c''(g)$ the policy in the lower left and upper right corner of g, respectively (see Figure 2).

We now sketch the general idea behind the algorithm for the pricing problem. Details will be filled in later.

Algorithm 2.

Initialization *Construct a grid of parallelograms (denoted by $\mathscr{G}$), such that each policy (s, S) with $s + S \leq \overline{s + S}$ is covered by at least one parallelogram $g \in \mathscr{G}$. Initialize the set of controls $\mathscr{C}' = \{c'(g) | g \in \mathscr{G}\} \cup \{c''(g) | g \in \mathscr{G}\}$. Thus, only controls in the corners of each parallelogram are initially considered.*

Step 1 *Update c^* such that $\forall c \in \mathscr{C}' : R(c^*) \leq R(c)$, where $R(c)$ is given by* (13).

Step 2 *For any $g \in \mathscr{G}$ for which $\underline{R}(c'(g); c''(g)) < R(c^*)$*

Refine parallelogram: *there might be policies c covered by g (but $c \notin \mathscr{C}'$) that could improve on c^*. Remove g from $\mathscr{G}$, and add to $\mathscr{G}$ a number of smaller parallelograms that together cover all policies originally covered by g. For each smaller parallelogram g' added in this way, add $c'(g')$ and $c''(g')$ to $\mathscr{C}'$.*

(Note that if $\underline{R}(c'(g); c''(g)) \geq R(c^*)$, then by Lemma 5 policies covered by g cannot improve on c^*. Consequently, these policies need not be evaluated.)

Step 3 *If any parallelogram was refined in Step 2, go to Step 1. Otherwise, terminate returning c^*. (Note that $\mathscr{G}$ and $\mathscr{C}'$ are stored for future calls to Step 1-3 with changed dual multipliers).*

This algorithm terminates, as parallelograms that need refining will become smaller and smaller until they cover only a single policy, at which point they need no further refining. (We find in Section 5 that the algorithm can avoid the evaluation of most policies.) Upon termination of the algorithm, c^* minimizes the reduced costs over all policies (s, S). ($\overline{s + S}$ is chosen such that the reduced costs of any policy with $s + S \geq \overline{s + S}$ exceeds the reduced costs of c^*.)

In the remainder of this section, we will fill in the details of each step. We first describe how certain intermediate calculations are stored, to be able to carry out Steps 1 and 2 efficiently. Then the construction of the grid is discussed in more detail. We discuss how to replace a parallelogram by smaller parallelograms. We discuss how to strengthen the lower bound. Finally, we discuss a procedure for setting $\overline{s + S}$.

4.1.1 Storing computations

We calculate and store m_k, $1/M_k$, $H(k-1,k)$ and $F^i(k-1,k)$ for $k \leq \overline{s+S}$ during initialization. $H(c)$ and $F^i(c)$ depend upon these stored values through (14). Note that

$$H(s-1,S)M_{S-s+1} = H(s,S)M_{S-s} + H(s-1,s)m_{S-s}, \tag{19}$$

and similar for F^i. For each policy (s,S) that is added to $\mathscr{C}'$, we calculate and store $H(s,S)M_{S-s}$ and $F^i(s,S)M_{S-s}, i \in \mathscr{I}$.

To this end, we first determine whether any policies (s',S) with $s' > s$ are already evaluated (using a suitable data structure). In that case, $H(s,S)$ can be calculated efficiently from $H(s',S)$ by repeated use of (19) (similar for $F^i(s,S)$). This significantly reduces computation times. For this reason, we will attempt to keep the number of different values S for which any policy needs to be evaluated limited.

When $O(c)$, $H(c)$ and $F^i(c)$ for $c \in \mathscr{C}'$ are required in Step 2 of Algorithm 1, and Step 1 and 2 of Algorithm 2, determining them thus requires only a single multiplication.

4.1.2 Grid construction

We have tested several methods for constructing a covering grid during initialization of Algorithm 2. We find that the algorithm is efficient regardless of the precise method that is used, as long as: 1. the grid is sufficiently sparse, and 2. the number of values for S for which any policies are initially added to $c \in \mathscr{C}'$ is kept limited.

We describe a method that we have found to have particularly robust performance across all parts. We determine first the values that will be used for S and $\Delta = S - s > 0$ in $\mathscr{C}'$. Take for instance $\{S_1, S_2, \ldots, S_N\} = \{0,1,2,3,4,6,8,12,16,24,32\ldots, \overline{s+S}\}$ (the step-size $S_{n+1} - S_n$ doubles whenever a power of two is reached). Use a similar range for $\{\Delta_1, \Delta_2, \ldots, \Delta_M\}$.

We now let

$$\bar{\mathscr{G}} = \{(g = (S_n - \Delta_{m+1}, S_n; S_{n+1} - \Delta_m, S_{n+1}) | n \in \{0, N-1\}, m \in \{0, M-1\}\} \tag{20}$$

We then let $\mathscr{G}$ consist of all parallelograms $g \in \bar{\mathscr{G}}$ that cover at least some policies (s,S) such that $-1 \leq s < S$ and $s + S < \overline{s+S}$. Figure 3 depicts parallelograms $g \in \mathscr{G}$ constructed in this manner, for the lower left area of the policy space that needs to be covered.

Note that the policy $(s',S') = c'(g)$ in the lower left corner of some $g \in \mathscr{G}$ will have $s' < -1$. As we assume $s \geq -1$, such policies need not be added to $\mathscr{C}'$. We consequently do not need to determine F^i and O, and we can suffice with determining a lower bound for the holding costs for policies with $s \geq -1$ that are covered by g. Note that by Lemma 4, the holding costs for such policies is bounded below by $H(S'(g)/2, S'(g)/2+1)$. For paralellograms with $s' < -1$, we adapt the evaluation of $\underline{R}(c'(g), c''(g))$ accordingly.

4.1.3 Refining a parallelogram

When a parallelogram $g = (s', S'; s'', S'')$ needs to be refined in Step 2 of Algorithm 4.1, we replace it by a number of smaller parallelograms covering the same policies. We consider two cases:

- $S' - s' \neq S'' - s'$. We replace the parallelogram by a number of parallelograms for which $S' - s' = S'' - s'$
- $S' - s' = S'' - s'$. The paralellogram is split into two parallelograms of (about) equal length.

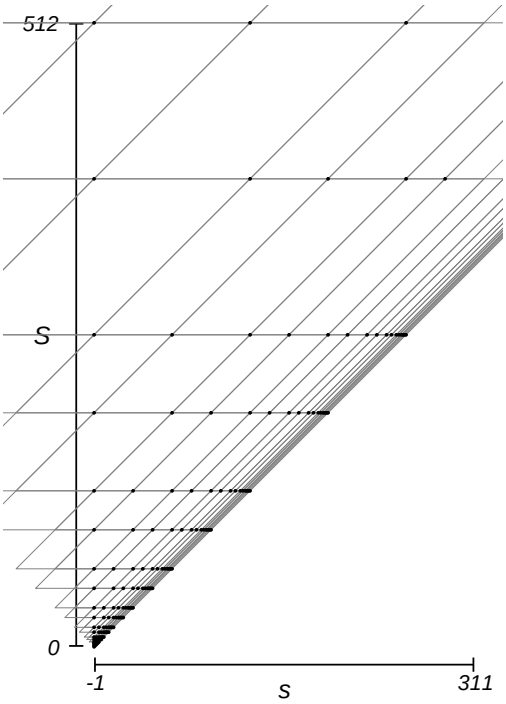

Figure 3: Some parallelograms $g \in \mathscr{G}$ after initial construction.

For graphical depiction of the manner in which we refine a parallelogram g for these two cases, see Figure 4. The reason for refining in this manner is that it limits the number of values for S for which policies need to be evaluated.

4.1.4 Strengthening the lower bound

A policy c is said to be integer feasible if

$$\forall i \in \mathscr{I} : F^i(c) \leq 1 - a^i. \tag{21}$$

Note that policies that do not satisfy this condition need to have a solution value 0 in any feasible *integer* solution. As such, by only considering policies satisfying (21) in Algorithm 1, we can strengthen the lower bound. Algorithm 2 can be easily adapted to return the *integer feasible* solution with lowest reduced costs. Note in particular that if for a parallelogram g the upper right corner $c''(g)$ is infeasible (does not satisfy (21)), then all policies covered by g are infeasible (by Lemma 2). As such, we need not consider refining g.

4.1.5 Bounding $s + S$

Note that any upper bound on $\min_{c \in \mathscr{C}} R(c) - \mu$ can be related to an upper bound $\overline{s+S}$ on $s + S$ for the minimizer of $R(\cdot)$ (through Lemma 4). If Algorithm 2 is used in a stand-alone manner to solve single-item inventory problems, $R(c) - \mu$ for any policy c can thus serve to determine $\overline{s+S}$.

When Algorithm 2 is used as part of Algorithm 1, no values for ν^i are available during initialization, and as such $R(c)$ cannot be determined. We can obtain suitable value for $\overline{s+S}$ using the following procedure:

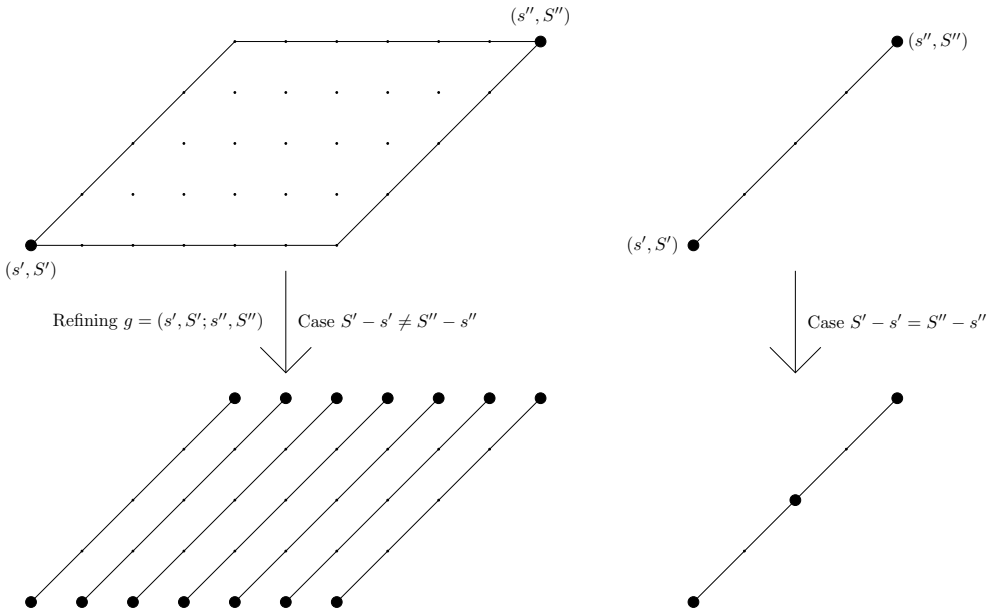

Figure 4: Depiction of the manner in which parallelograms are refined in the algorithms.

1. Determine $\tilde{s}$ such that $\forall i \in \mathscr{I} : F^i(\tilde{s}, \tilde{s}+1) \leq \frac{\epsilon(1-a^i)}{|\mathscr{J}^i|}$. Use e.g. $\epsilon = 10^{-6}$. Note that $\tilde{s}$ provides sufficient safety stock to virtually guarantee no stock-outs for the part.
2. For any $\tilde{S} > \tilde{s}$, $H(\tilde{s}, \tilde{S}) + O(\tilde{S} - \tilde{s})$ can be used as an upper bound on $\min_{c \in \mathscr{C}} R(c) - \mu$. We then determine y such that $H(y-1, y) > H(\tilde{s}, \tilde{S}) + O(\tilde{S} - \tilde{s})$ and set $\overline{s+S} = 2y$.

$H(\tilde{s}, \tilde{S}) + O(\tilde{S} - \tilde{s})$ has proven a valid upper bound on $R - \mu$ for all parts in a wide range of test problems.

It is conceivable (but highly unlikely) that in some cases $R(c^*) - \mu > H(\tilde{s}, \tilde{S}) + O(\tilde{S} - \tilde{s})$ after execution of the algorithm, indicating that the bound is not valid. In that case, we can simply determine a new value for $\overline{s+S}$ based on $R(c^*) - \mu$. We then expand the grid to cover all policies (s, S) with $s + S$ lower than the new value for $\overline{s+S}$, and continue at Step 2 of Algorithm 2. With this modification, the resulting algorithm is *guaranteed* to terminate at the optimum.

4.2 Rounding heuristic

We only have obtained conclusive results for one simple rounding heuristic yet. It simply selects for each part the active policy with the highest costs in the objective function (5). A policy $c \in \mathscr{C}'$ is said to be active for part $j \in \mathscr{J}$ if the primal decision variable x_{jc} associated with j and c is strictly positive.

While this procedure is not guaranteed to give a feasible solution, for all tested problems it did give a feasible solution.

5 Results

The study was performed in close collaboration with a repair shop of aircraft components. We have obtained a data set from this repair shop. In discussion with the managers, we have used

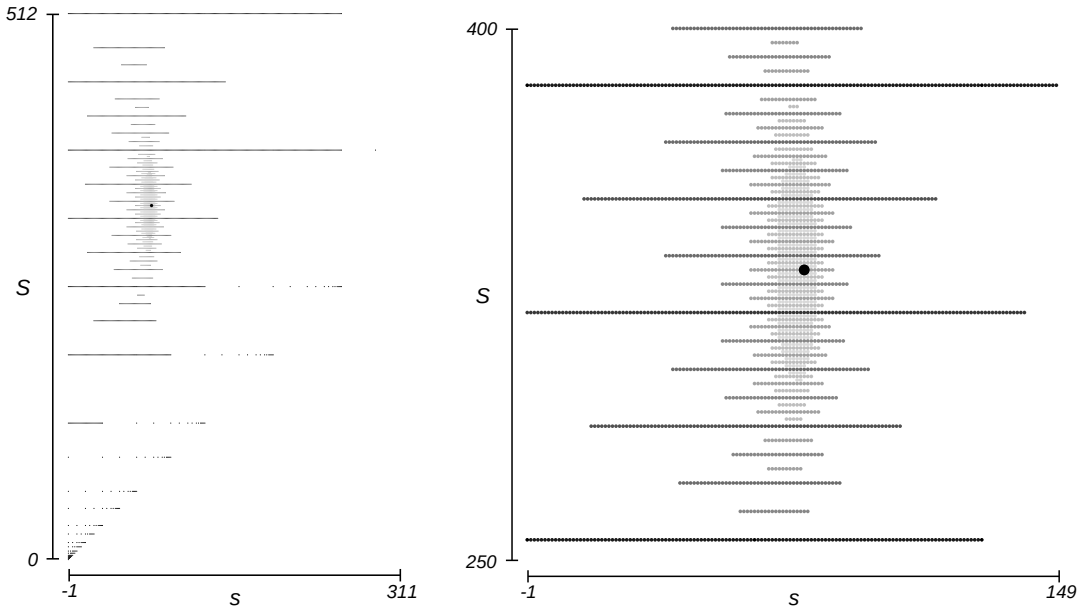

Figure 5: In the left part of the figure, the lower left area of the s, S space that needs to be covered is depicted. Dots represent policies that are evaluated during execution of Algorithm 2. Darker/lighter dots represent policies that are added in earlier/later iterations of Step 2 of the algorithm. The large dark dot represents the policy with the lowest reduced costs. On the right side of the figure, a detail of the left side is depicted.

this dataset to construct an optimization problem that reflects the situation at the company. This problem consists of about 4000 spare part types and 500 components types. Lead times and prices were obtained from the ERP system. For the purpose of this study, estimates for λ^i and Y_j^i were obtained using historical data on repair volumes and on the usages of spare parts in the repairs. The fill rate targets for the repair of each component type were set based on discussions with the management of the repair shop.

We use Algorithm 1 to solve the resulting optimization problem. The solution to the continuous relaxation was found in minutes on a modern computer.

Algorithm 2 was very efficient at keeping the number of controls to be evaluated limited. In total, of all policies (s, S) with $s+S$ smaller than the upper bound, only 0.3% need to be evaluated. This is about 200 policies per part on average. Moreover, the algorithm is quite effective in limiting the number of different values for S for which any policies need to be evaluated (for the importance of this, see the discussion in Section 4.1.1). The average number of values for S for which any policies need to be considered is 35.

Figures (5) illustrate which policies are evaluated for a typical high demand part. In most of the solution space, the sparse grid that was initially contructed suffices to establish optimality. Lemma 5 thus provides an effective lower bound.

In the area around the optimal policy, the algorithm typically evaluates all policies because it needs a fine grid in order to establish optimality (see Figure 5). The dual multipliers generally exhibit only small changes when re-solving the continuous relaxation after adding policies to $\mathscr{C}_j$ for each part (Step 2 of 1). Because we store the grid $\mathscr{G}$ and the evaluated policies $\mathscr{C}$ for each part, Algorithm 1 terminates already after a few (typically < 5) iterations. Evaluation of policies around

the optimum in the optimum is thus beneficial in terms of efficiency.

Because relatively little policies need to be evaluated, the memory requirements for storing the values of all solutions remain within bounds. (Memory requirements were still significant, but not prohibitively so: the algorithm uses about 300MB memory in total. Most memory is used for storing the intermediate calculations in the pricing algorithm.)

The proposed rounding heuristic finds a feasible solution instantly. We find a gap between continuous relaxation and the IP solution of about 6%. It can be argued that such a gap is good enough for practical purposes. Preliminary experiments with other rounding heuristics indicate that it is possible to decrease the gap significantly, while keeping the computation time within limits. (We do not expect to *close* the gap, as the problem is NP-hard.) We can thus confidently state that the algorithm proposed in this paper is practable to solve real-life systems.

References

AFM: 2009, MRO world forecast report, *Airline Fleet Management* **64**, 38–48.

Agrawal, N. and Cohen, M. A.: 2001, Optimal material control in an assembly system with component commonality, *Naval Research Logistics* **48**, 409–429.

Akçay, Y. and Xu, S. H.: 2004, Joint inventory replenishment and component allocation optimization in an assemble-to-order system, *Management Science* **50**, 99–116.

Axsäter, S.: 2006, *Inventory Control*, 2nd edn, Springer.

Benjaafar, S. and ElHafsi, M.: 2006, Production and inventory control of a single product assemble-to-order system with multiple customer classes, *Management Science* **52**, 1896–1912.

Cheng, F., Ettl, M., Lin, G. and Yao, D. D.: 2002, Inventory-service optimization in configure-to-order systems, *Manufacturing & Service Operations Management* **4**, 114–132.

Cheung, K. L. and Hausman, W.: 1995, Multiple failures in a multi-item spare inventory model, *IIE transactions* **27**, 171–180.

Dayanik, S., Song, J.-S. and Xu, S. H.: 2003, The effectiveness of several performance bounds for capacitated production, partial-order-service, assemble-to-order systems, *Manufacturing & Service Operations Management* **5**, 230–251.

Gallien, J. and Wein, L. M.: 2001, A simple and effective component procurement policy for stochastic assembly systems, *Queueing Systems* **38**, 221–248.

Glasserman, P. and Wang, Y.: 1998, Leadtime-inventory trade-offs in assemble-to-order systems, *Operations Research* **46**, 858–871.

Güllü, R. and Köksalan, M.: 2009, A model for performance evaluation and stock optimization in a kit management problem. Paper presented at the 16th ISIR symposium.

Hausman, W. H., Lee, H. L. and Zhang, A. X.: 1998, Joint demand fulfillment probability in a multi-item inventory system with independent order-up-to policies, *European Journal of Operational Research* **109**, 646–659.

Kranenburg, A. and van Houtum, G.: 2008, Service differentiation in spare parts inventory management, *Journal of the operations research society* **59**, 946–955.

Kranenburg, A. and van Houtum, G.: 2009, A new partial pooling structure for spare parts networks, *European Journal of Operational Research* **199**, 908–921.

Lu, Y. and Song, J.-S.: 2005, Order-based cost optimization in assemble-to-order systems, *Operations Research* **53**, 151–169.

Lu, Y., Song, J.-S. and Yao, D. D.: 2003, Order fill rate, leadtime variability, and advance demand information in an assemble-to-order system, *Operations Research* **51**, 292–308.

Lu, Y., Song, J.-S. and Yao, D. D.: 2005, Backorder minimization in multiproduct assemble-to-order systems, *IIE Transactions* **37**, 763–774.

Lu, Y., Song, J.-S. and Zhao, Y.: 2010, No-holdback allocation rules for continuous-time assemble-to-order systems, *Operations Research* **58**, 691–705.

Rosling, K.: 1989, Optimal inventory policies for assembly systems under random demands, *Operations Research* **37**, 565–579.

Song, J.-S.: 1998, On the order fill rate in a multi-item, base-stock inventory system, *Operations Research* **46**, 831–845.

Song, J.-S.: 2000, A note on assemble-to-order systems with batch ordering, *Management Science* **46**, 739–743.

Song, J.-S.: 2002, Order-based backorders and their implications in multi-item inventory systems, *Management Science* **48**, 499–516.

Song, J.-S., Xu, S. H. and Liu, B.: 1999, Order-fulfillment performance measures in an assemble-to-order system with stochastic leadtimes, *Operations Research* **47**, 131–149.

Song, J.-S. and Yao, D. D.: 2002, Performance analysis and optimization of assemble-to-order systems with random lead times, *Operations Research* **50**, 889–903.

Song, J.-S. and Zipkin, P.: 2003, Assemble-to-order systems, *in* A. G. de Kok and S. C. Graves (eds), *Supply chain management: design, coordination and operation*, Vol. 11 of *Handbooks in operations research and management science*, Elsevier, North-Holland, The Netherlands, pp. 516–596.

Svoronos, A. and Zipkin, P.: 1991, Evaluation of one-for-one replenishment policies for multiechelon inventory systems, *Management Science* **37**, 68–83.

Zhang, A. X.: 1997, Demand fulfillment rates in an assemble-to-order system with multiple products and dependent demands, *Production and Operations Management* **6**, 309–324.

Zhao, Y.: 2009, Analysis and evaluation of an assemble-to-order system with batch ordering policy and compound poisson demand, *European Journal of Operational Research* **198**, 800–809.

Zhao, Y. and Simchi-Levi, D.: 2006, Performance analysis and evaluation of assemble-to-order systems with stochastic sequential lead times, *Operations Research* **54**, 706–724.

Zheng, Y. and Federgruen, A.: 1991, Finding optimal (s, S) policies is about as simple as evaluating a single policy, *Operations research* **39**, 654–665.

Zipkin, P.: 1986, Stochastic leadtimes in continuous-time inventory models, *Naval Research Logistics Quarterly* **33**, 763–774.

A Proof of Lemma 4

Let $j \in \mathscr{J}$ be given and let $(s, S) \in \mathscr{C}_j$ be the policy to control the inventory for spare part j. We will prove that

$$H_j(s, S) \geq H_j\left(\frac{s+S-1}{2}, \frac{s+S+1}{2}\right),$$

where $H_j(s, S)$ is the holding cost rate for part j.

We will apply renewal theory to compute the left hand side. Every time a replenishment order is placed, the inventory position is raised to S. From that moment on, the inventory decreases until it drops to or below s. On that time a new order is placed, the inventory position is raised to S again and the process repeats itself. This process can be viewed as a renewal reward process. Define $C(s, S)$ as the expected holding cost and $T(s, S)$ as the expected time during a cycle. For simplicity of notation, define $p_d = \mathbf{P}(D = d)$ for $d \in \mathbf{N}$ as the compounding distribution for the demand for spare part j. It then holds, that

$$C(s, S) = \frac{1}{\lambda_j} H_j(S-1, S) + \sum_{d=1}^{S-s-1} p_d C(s, S-d), \tag{22}$$

$$T(s, S) = \frac{1}{\lambda_j} + \sum_{d=1}^{S-s-1} p_d T(s, S-d). \tag{23}$$

To see why these formulas are correct, note that the inventory position is S precisely after an order is placed. It will remain at that level until there is demand D for the spare part. If $D > S-s-1$, the inventory position will drop below to or below s and a new order is placed, completing the cycle. Otherwise, the inventory position will be $S - D > s$. During the remainder of the cycle, the inventory position behaves as if inventory was controlled by an $(s, S-D)$ policy. This explains the second term in the right hand side of (22) and (23).

The elementary renewal theorem now states that

$$H_j(s, S) = \frac{C(s, S)}{T(s, S)} = \frac{\lambda_j C(s, S)}{\lambda_j T(s, S)}.$$

By this equation, we can assume $\lambda_j = 1$ without loss of generality.

To ease the notation, define

$$v : \mathbf{N} \to \mathbf{R} : n \mapsto v(n) = H_j\left(s + \frac{n}{2}, s + 1 + \frac{n}{2}\right).$$

$H_j(s - \frac{1}{2}, s + \frac{1}{2})$ is defined as the average of $H_j(s-1, s)$ and $H_j(s, s+1)$ for all $s \in \mathbf{Z}$. Convexity of $s \mapsto H_j(s, s+1)$ then implies that

$$H_j(i+1, i+2) - H_j(i, i+1) \geq H_j(i, i+1) - H_j(i-1, i)$$

for all $i \in \frac{1}{2}\mathbf{Z}$. The function v therefore satisfies

$$v(n+1) - v(n) \geq v(n) - v(n-1); \tag{24}$$

i.e., the function v is convex. For later convenience, we also introduce

$$\bar{v} : \mathbf{N}_0^2 \to \mathbf{R} : (i, j) \mapsto \bar{v}(i, j) = v(i) - v(j).$$

(24) implies that the function $i \mapsto \bar{v}(i+1,i)$ is increasing. This implies that $\bar{v}(i+1,i) \leq \bar{v}(j+1,j)$ whenever $i \leq j$. It follows for $n, j \in \mathbf{N}_0$ with $1 \leq j \leq n$, that

$$\begin{aligned}\bar{v}(n,n-j) &= v\,(n) - v\,(n-j) \\ &= \sum_{k=0}^{j-1} \bar{v}(n-k,n-k-1) \leq \sum_{k=0}^{j-1} \bar{v}(n,n-1) = j\bar{v}(n,n-1).\end{aligned}$$

As the left and right hand side are obviously equal for $j = 0$, we conclude for $0 \leq j \leq n$, that

$$\bar{v}(n,n-j) \leq j\bar{v}(n,n-1). \tag{25}$$

We will prove the lemma for fixed s by induction on $S > s$. Note that for fixed s, the policy (s,S) is alternatively characterized by the difference $S - s - 1$. To ease the notation, and to clarify the inductive argument, we define $N = S - s - 1$ and apply the identification $N \sim (s, s+1+N) \in \mathscr{C}_j$. Note that $S > s$ is equivalent to $N \geq 0$. We will apply the following lemma.

Lemma 6. *For all $N \in \boldsymbol{N}$,*

$$\sum_{j=1}^{N} jp_jT(N-j) \leq N.$$

Proof. For $N = 1$ the result follows from the observation that $p_1 \leq 1$. Assume now that it holds for all $n < N$. Then

$$\begin{aligned}\sum_{j=1}^{N} jp_jT(N-j) &= Np_N + \sum_{j=1}^{N-1} jp_jT(N-j) \\ &= Np_N + \sum_{j=1}^{N-1} jp_j \left(1 + \sum_{k=1}^{N-j} p_kT(N-j-k)\right) \\ &= \sum_{j=1}^{N} jp_j + \sum_{k=1}^{N-1} p_k \sum_{j=1}^{N-k} jp_jT(N-k-j) \\ &\leq \sum_{j=1}^{N} jp_j + \sum_{k=1}^{N-1} p_k(N-k) = \sum_{j=1}^{N} Np_j = N,\end{aligned}$$

where the inequality relies on the induction hypothesis. This proves the claim. □

We are now ready to prove Lemma 4. Recall that $N = S - s - 1$, so

$$v(N) = H_j\left(\frac{2s+N}{2}, \frac{2s+2+N}{2}\right) = H_j\left(\frac{s+S-1}{2}, \frac{s+S+1}{2}\right).$$

Similarly $v(2N) = H_j(S-1,S)$. Using the notation introduced in this appendix, we should thus prove the following.

Lemma 7. *$C(N)/T(N) \geq v\,(N)$ for all $N \in \boldsymbol{N}_0$.*

Proof. For $N = 0$ the result follows by definition. Assume now that it holds for all $n < N$. We then have for all $1 \leq j \leq N$

$$0 \leq C(N-j) - v\,(N-j)\,T(N-j).$$

Multiplying this expression by p_j and summing it from $j = 1$ to N, we obtain

$$0 \leq \sum_{j=1}^{N} p_j C(N-j) - \sum_{j=1}^{N} p_j v\,(N-j)\,T(N-j). \tag{26}$$

Applying (25), Lemma 6 and monotonicity of $i \mapsto \bar{v}(i+1,i)$, it follows for all $1 \leq j \leq N$, that

$$\begin{aligned}\sum_{j=1}^{N} p_j \bar{v}(N,N-j)T(N-j) &\leq \bar{v}(N,N-1)\sum_{j=1}^{N} j p_j T(N-j)\\ &\leq N\bar{v}(N,N-1)\\ &\leq \sum_{k=0}^{N-1} \bar{v}(2N-k,2N-k-1) = v(2N) - v(N).\end{aligned}$$

Rearranging terms, we can rewrite this as

$$0 \leq v(2N) - v(N) + \sum_{j=1}^{N} p_j v(N-j)T(N-j) - \sum_{j=1}^{N} p_j v(N)T(N-j). \tag{27}$$

Adding (26) and (27), we obtain

$$0 \leq v(2N) + \sum_{j=1}^{N} p_j C(N-j) - v(N)\left(1 + \sum_{j=1}^{N} p_j T(N-j)\right) = C(N) - v\,(N)\,T(N).$$

This proves the claim. □