An iterative method for the simultaneous optimization of repair decisions and spare parts stocks

R.J.I. Basten • M.C. van der Heijden • J.M.J. Schutten

November 3, 2009

Abstract

In the development process of a capital good, it should be decided how to maintain it once it is in the field. The level of repair analysis (LORA) is used to answer the questions: 1) which components to repair upon failure, and which to discard, 2) at which locations in the repair network to perform the repairs, and 3) at which locations to deploy resources, such as repair equipment. Next, it should be decided what amount of spare parts to store at each location in the network in order to guarantee a certain availability of the product. Usually, the LORA and the spare parts stocking problem are solved sequentially. However, solving the LORA first can lead to high spare parts costs. Therefore, we propose an iterative approach to solve the two problems jointly. We find that the total costs are lowered with 3.2% on average and almost 35% at maximum in our experiments. A cost reduction of a few percent may be worth hundreds of thousands of euros over the life cycle of a capital good.

Keywords: Service logistics, Level of repair analysis, Spare parts, Inventories

1 Introduction

Manufactured products and installations are often prone to failure. Since capital goods are expensive, they will be repaired upon failure. We concentrate on capital goods that have high downtime costs, examples of which are manufacturing equipment, defence systems, medical devices, and airplanes. Instead of focusing on the initial price, customers purchasing capital goods increasingly take the total life cycle costs (LCC) into account (Ferrin and Plank, 2002). We also observe a trend in which customers outsource activities for product upkeep to the original equipment manufacturer (OEM), using service contracts that guarantee a certain service level against fixed annual costs. Therefore, OEMs need methods to estimate (and influence) the LCC of newly designed capital goods, especially the costs of corrective maintenance.

Quick recovery of a capital good upon failure is important, which is why it is typically *repaired by replacement* of a defective component by a functioning spare part. If no spare part is available, the capital good is down waiting for spares. In defence, the components that are taken out of the product are called *LRUs* or *line replaceable units*. Defective LRUs can either be discarded or repaired. Since an LRU may be very expensive, it is typically repaired, usually by replacement of a subcomponent. Since such a replacement is typically performed in a repair shop, these subcomponents are called *SRUs* or *shop replaceable units*. The SRU should in turn be repaired, possibly by replacement of a *part*, or discarded. The product is thus characterized by a *multi-indenture product structure* as shown in Figure 1. In general terms, we use the term *components*, which can have *subcomponents*.

The installed base, consisting of all systems that are sold and still in use, is usually dispersed over a large geographical area. For system upkeep, a support network is required with facilities that are close to the installed base. However, locating spares and test equipment at each operating site is usually expensive. Therefore, often also more central locations are used to stock spare parts and to locate expensive test equipment. As a consequence, a repair network usually consists of multiple *echelon levels*. Figure 2

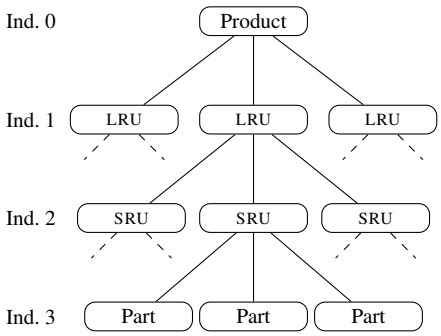

Figure 1: Product structure

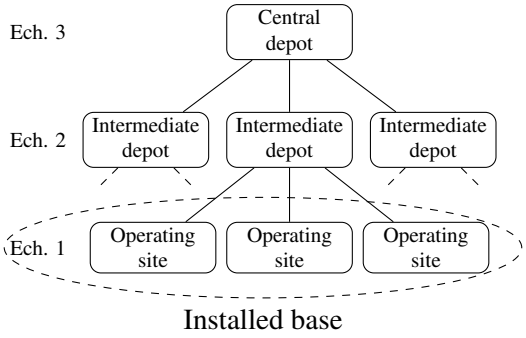

Figure 2: Repair network

shows an example including the naming convention that we use. If there are multiple echelon levels, it should be decided where to perform repairs, where to locate resources (e.g., test and repair equipment), and where to stock spare parts.

The *level of repair analysis problem* (LORA) is to determine whether a component should be repaired or discarded upon its failure, and at which location in the repair network to perform repairs. To enable certain repairs, resources have to be installed. The goal is to achieve the lowest LCC, consisting of both fixed costs (e.g., test equipment and tools) and costs that are variable in the number of failures (e.g., hiring service engineers and transportation of components). Given the LORA decisions, the *spare parts stocking problem* is to allocate spare parts inventory in a repair network such that a certain availability of the installed base is achieved against the lowest possible spare parts costs. Since we are interested in the delay time due to a lack of spares, the availability refers to the *supply availability*, which is defined as $\frac{\text{MTBM}}{\text{MTBM}+\text{MTWS}}$, with MTBM being the mean time between maintenance and MTWS being the mean time waiting for spares.

Although sequentially solving the LORA and the spare parts stocking problem yields a solution that achieves the target availability, the total costs may be too high: the LORA often results in performing many repairs at the most central location, since that leads to locating repair equipment at one location only. However, this implies that the installed base faces long repair lead times, which increases the need for spare part inventories. If repairs are performed at the operating sites, more repair equipment is needed, but less spares. We conclude that a sequential approach may lead to an excessive spare parts investment, which is especially harmful if the spare parts costs are a major component of the total costs. We encountered such a situation in the case study that we performed at Thales Nederland, a manufacturer of naval sensors and naval command and control systems. In an integrated model, the higher costs of resources can be balanced against the lower costs of spares.

This paper is organised as follows. In Section 2, we discuss related literature and show the contribution of our paper. We outline our integrated model for LORA and spare parts stocking in Section 3. In Section 4, we explain the general idea behind our iterative method to solve this model. Next, we specify a basic algorithm and two variants in Section 5. In Section 6, we examine the added value of our iterative approach in a numerical experiment. Also, we identify the variant of our iterative algorithm that performs best in terms of cost reduction. We apply this variant in Section 7 to our case study at Thales Nederland. We give our conclusions and recommendations for further research in Section 8.

2 Literature review

Three streams of literature are relevant here: the literature on LORA, spare parts stocking, and the joint problem of LORA and spare parts stocking, which we discuss in Sections 2.1 to 2.3, respectively. Next, we show our contribution in Section 2.4.

2.1 Level of repair analysis

The LORA problem is generally modelled as a mixed integer linear programming model which is solved using a commercial solver (e.g., CPLEX). Exceptions are Barros and Riley (2001), who use a branch-and-bound approach to solve the model, and Saranga and Dinesh Kumar (2006), who use a genetic algorithm.

Barros (1998), Barros and Riley (2001), Saranga and Dinesh Kumar (2006), and Basten et al. (2009) assume infinite capacity of resources and aggregate all data per echelon level. The first assumption means that it is unneccessary to have two resources of the same type at one location and that repair times are not influenced by the workload. The second assumption means that the same decisions are taken at each location at one echelon level. The key difference between these papers lies in the restrictions on the component-resource relations. Barros (1998) and Barros and Riley (2001) assume that all components at one indenture level require the same resource in order to be repaired and Saranga and Dinesh Kumar (2006) assume that each component requires exactly one resource, which is required by that component only. In contrast, Basten et al. (2009) allow for components requiring multiple resources and multiple components requiring the same resource.

Basten et al. (2008) use similar assumptions as Basten et al. (2009), except that the former do not aggregate all data per echelon level. The authors show that the LORA problem can be modelled efficiently as a generalized minimum cost flow model. Brick and Uchoa (2009) also explicitly model the repair network, but consider one echelon level only. Furthermore, the authors effectively assume two indenture levels. Integrated in their LORA is the decision of which facilities to open.

2.2 Spare parts stocking

A vast amount of literature exists on the multi-item spare parts stocking problem. We are interested in expensive, slow moving, repairable components. The paper of Sherbrooke (1968) is generally seen as the seminal paper in this field. He developed the METRIC model (Multi-Echelon Technique for Recoverable Item Control), which is the basis for a huge stream of METRIC type models. The METRIC type models aim to find the most cost effective allocation of spare parts in a network that achieves a target supply availability of the installed base. We refer to Sherbrooke (2004) and Muckstadt (2005) for an extensive overview of the literature on METRIC type models.

2.3 Joint problem of level of repair analysis and spare parts stocking

To our knowledge, Alfredsson (1997) is the only one who presented an approach to solve the joint problem of LORA and spare parts stocking, assuming a single-indenture product structure and a two-echelon repair network. Each component requires one specific tester (resource), which is required by one component only. Furthermore, one multi-tester exists. This multi-tester can be used for the repair of one component and adapters can be added in a fixed order to enable the multi-tester to be used for the repair of additional components. If the multi-tester can be used to repair a component, the original specific resource for that component is not used anymore. Resources are capacitated, which means that multiple resources of the same type may be required at one location. System downtime includes the waiting times for the resources, the repair times, and the waiting times for spares. The problem is modelled as a non-linear integer programming model and Alfredsson uses a decomposition method that sequentially decomposes the overall problem in smaller subproblems.

2.4 Contribution

We develop a new method to solve the joint problem of LORA and spare parts stocking. Our model covers two practically relevant features that the model of Alfredsson (1997) does not cover:

- general resource-component relations, which means that a component may require multiple resource simultaneously and that resources may be required by multiple components;
- no theoretical limit on the number of echelon levels and indenture levels.

We show that our algorithm improves the sequential approach with a maximum cost reduction of over 35% and over 3% on average, which may be worth hundreds of thousands of euros over the life cycle of a capital good. We also show the added value of our approach using a case study at Thales Nederland, in which we achieve a cost reduction of almost 10%.

3 Model

In this section, we outline the model that we propose. The general problem description is presented in Section 3.1, and Section 3.2 lists the assumptions. We give the mathematical model formulation in Section 3.3. In Section 3.4 we make two further assumptions. These are not critical for our approach, but they ease the presentation in the remainder of this paper and decrease the problem size.

3.1 Problem description

We aim to solve the integrated model of LORA and spare parts stocking. This means that given a product design, an installed base, and a repair network, we decide on:

- which components to repair upon failure, and which to discard,
- the repair location in the network for each component that we decide to repair,
- the location(s) in the network where we install resources (e.g. repair equipment), and
- the locations and amounts of spare parts to stock for each component.

We model the LORA part of the problem similar to the model of Basten et al. (2008); at each location there are three possible decisions for each component: repair, discard, and move to the next higher echelon level (if such an echelon level exists). If we discard a component and purchase a new one, no further decisions need to be taken for this component or its subcomponents. If we repair a component and find out that the failure is due to the failure in a subcomponent, a decision needs to be taken for that subcomponent, at the same location. If we can repair the component without replacing a subcomponent, no further decisions need to be taken. If we move a component to the location at the next higher echelon level, a decision needs to be taken for that component at that level. The decision that a certain component is to be repaired, discarded, or moved at a certain location, means that the required resources (if any) should be installed at that location as well. In order to reduce downtime, we also require spare parts inventories at the location where a component is exchanged or upstream in the network. Our goal is to achieve the lowest possible LCC, subject to a constraint on the availability. The LCC include the variable and fixed LORA costs, as mentioned in Section 1, and the spare parts holding costs.

3.2 Assumptions

Similar to what is assumed in the standard VARI-METRIC model (see, e.g., Muckstadt, 2005; Sherbrooke, 2004), we assume that:

- the number of failures in a time period of fixed length is Poisson distributed with constant rate;
- a failure in a component with subcomponents is due to a failure in at most one subcomponent;
- a location in the repair network at echelon level e is only supplied with functioning spare parts from its parent-location at echelon level $e+1$, not by a lateral supply from another location at echelon level e or by emergency shipments from locations at an echelon level $> e+1$;
- one for one replenishment (or an $(s-1,s)$ inventory control policy) is appropriate for every component at every echelon level.

In addition to the assumptions above, we assume that at all locations at one echelon level we take the same LORA decisions (the repair/discard decisions and the location of resources); we call this symmetrical LORA decisions. We further assume that resources have infinite capacity, so that at each echelon level with n locations, we may locate either zero or n resources. The locations and amounts of spare parts stocks need not be symmetrical, and the repair networks that we consider need not be symmetrical either. For example, the number of operating sites per intermediate depot may differ for the various intermediate depots, or repair costs may not be equal at all operating sites. In the latter case, we use the weighted average repair costs as the repair costs at echelon level 1. In symmetrical repair networks, the optimal solution consists of symmetrical LORA decisions; in asymmetrical networks this is not necessarily true. Allowing for non-symmetrical LORA decisions in the model is straightforward, but the method to solve the model is somewhat more complicated (see also Section 8).

3.3 Mathematical model

In Sections 3.3.1 to 3.3.3, we introduce the sets that we require in our model, the decision variables, and the input parameters, respectively. Next, we give the mathematical model formulation in Section 3.3.4.

3.3.1 Sets

Let the set C consist of all components in the product structure and let $C_1 \subseteq C$ be the set of LRUs. A component (parent) may contain subcomponents (children), which are at the next higher indenture level. Γ_c denotes the set of children of component c. This set may be empty.

We model a multi-echelon repair network, with L being the set of locations and E being the set of echelon levels. In general, there are three possible decisions $d \in D$ to take for each component $c \in C$:

- Discard: component c is scrapped and a new one is acquired.
- Repair: component c is repaired, possibly by replacing a defective child $b \in \Gamma_c$ with a functioning one.
- Move: component c is moved to echelon level $e+1$.

The set D_e consists of the decisions that are available at echelon level e. So, for all $e \in E \setminus e^{\text{CEN}}$: $D_e = \{\text{discard}, \text{repair}, \text{move}\}$ and $D_{e^{\text{CEN}}} = \{\text{discard}, \text{repair}\}$.

Finally, we model resources $r \in R$. Let Ω_r consist of all combinations of a component c and a decision d for which resource r is required. So, if component c requires resource r in order to enable decision d, then the 2-tuple $(c,d) \in \Omega_r$. One tuple may be an element of multiple sets Ω_r, indicating that more resources are required simultaneously.

3.3.2 Decision variables

There are three sets of decisions variables:

$$X_{c,e,d} = \begin{cases} 1, & \text{if for component } c \in C \text{ at echelon level } e \in E \text{ decision } d \in D_e \text{ is taken} \\ 0, & \text{otherwise;} \end{cases}$$

$$Y_{r,e} = \begin{cases} 1, & \text{if resource } r \text{ is located at echelon level } e \\ 0, & \text{otherwise;} \end{cases}$$

$$S_{c,l} = \text{the number of spare parts of component } c \text{ located at location } l.$$

In addition, $\mathscr{X}$ denotes the matrix of all variables $X_{c,e,d}$ and $\mathscr{S}$ denotes the matrix of all variables $S_{c,l}$.

3.3.3 Input parameters

Let λ_c be the total annual failure rate for component $c \in C$ over all operating sites. Since a failure in a component is due to a failure in at most one subcomponent, it should hold that $\sum_{b \in \Gamma_c} \lambda_b \leq \lambda_c, \ \forall c \in$

C. $vc_{c,e,d}$ are the variable costs of taking action d for component c at echelon level e. Without loss of generality, we have chosen to minimize the average total costs per year with our definition of λ_c. Therefore, we define $fc_{r,e}$ to be the annual fixed costs to locate resource r at echelon level e. hc_c is the annual costs of holding one spare of component c.

3.3.4 Mathematical model formulation

The following model defines our problem:

$$\text{minimize} \sum_{c\in C}\sum_{e\in E}\sum_{d\in D} vc_{c,e,d}\cdot\lambda_c\cdot X_{c,e,d} + \sum_{r\in R}\sum_{e\in E} fc_{r,e}\cdot Y_{r,e} + \sum_{c\in C}\sum_{l\in L} hc_c\cdot S_{c,l} \tag{1}$$

subject to:

$$\sum_{d\in D_1} X_{c,1,d} = 1\ ,\forall c\in C_1 \tag{2}$$

$$X_{c,e,\text{move}} \le \sum_{d\in D_{e+1}} X_{c,e+1,d}\ ,\forall c\in C\ ,\forall e\in E\setminus e^{\text{cen}} \tag{3}$$

$$X_{c,e,\text{repair}} \le \sum_{d\in D_e} X_{b,e,d}\ ,\forall c\in C\ ,\forall b\in\Gamma_c\ ,\forall e\in E \tag{4}$$

$$X_{c,e,d} \le Y_{r,e}\ ,\forall r\in R\ ,\forall(c,d)\in\Omega_r\ ,\forall e\in E \tag{5}$$

$$\text{availability}(\mathscr{X},\mathscr{S}) \ge \text{target availability} \tag{6}$$

$$X_{c,e,d}, Y_{r,e} \in \{0,1\} \tag{7}$$

$$S_{c,l} \in \mathbb{N} \tag{8}$$

Constraint 2 guarantees that a decision is taken for each of the failures that occur in LRUs at the operating sites. Constraint 3 makes sure that if the decision is taken to move a component to the next higher echelon level, a decision is taken at that echelon level. Constraint 4 assures that if the decision is taken to repair a component, a decision is taken for all its subcomponents as well. Constraint 5 guarantees that repair, discard, or move actions are only performed at echelon levels at which all required resources are available. Constraints 2 to 5 are the 'LORA constraints'. Constraint 6 is the 'spare parts stocking constraint'; it makes sure that the target availability is met. Together with the third (and last) term in the objective function, it makes up the spare parts stocking problem. However, the availability is a non-linear function of the repair/discard decisions and the spare parts decisions. Therefore, this mathematical model cannot be solved in this form using an ILP solver, but is used to define the problem clearly. To calculate the availability, we use the VARI-METRIC model. In these calculations, we use the exact structure of the repair network, e.g., including failure rates per operating site.

3.4 Demarcation

We make two additional assumptions that are not critical for our approach, but ease the presentation in the remainder of this paper and decrease the problem size.

- We only consider resources that are required to enable the repair option; resources that are required for discard and move do not occur frequently in practice (e.g., not in the case study).
- Discard costs are equal at all echelon levels, which is a reasonable assumption since the main part of the discard costs consists of the costs of acquiring a replacement component. As a result, we consider discard at the highest echelon level only.

4 General approach

We first discuss the current way of working in Section 4.1, then we discuss the iterative approach in Section 4.2, and we focus on the feedback mechanism that is part of the iterative approach in Section 4.3.

4.1 Sequential approach

The basic way of executing a LORA and spare parts stocking analysis is sequentially. First, a LORA is performed. The goal is to achieve the lowest possible life cycle costs, consisting of both fixed costs ($\sum_{r\in R}\sum_{e\in E} fc_{r,e}\cdot Y_{r,e}$), and costs that vary with the number of failures ($\sum_{c\in C}\sum_{e\in E}\sum_{d\in D} vc_{c,e,d}\cdot\lambda_c\cdot X_{c,e,d}$). The spare parts holding costs are sometimes included (see, e.g., Saranga and Dinesh Kumar, 2006), but it is not clear how these costs should be estimated. The reason is that the number of spare parts that should be stocked as a result of a repair/discard decision is not only related to the repair/discard decision for the component itself, but also to the decisions for other components. For example, assume that an availability of 95% should be achieved for a product consisting of two components (A and B), then an unavailability of at maximum 5% is allowed for the two components together. If component A is relatively inexpensive, it may be optimal to achieve an unavailability of 1% for component A, and an unavailability of 4% for component B. If the repair/discard decision for component B is changed, so that the lead time for that component is reduced, it may be optimal to achieve an unavailability of 3% for component B and an unavailability of 2% for component A.

As a result, we see in practice, e.g., at Thales Nederland, that spare parts holding costs are not included in the LORA costs. Instead, given the decisions that result from the LORA, a spare parts stocking problem is solved (e.g., using a METRIC type method) that determines where to locate spare parts in the repair network, such that a target availability of the installed base is achieved against the lowest possible spare parts holding costs ($\sum_{c\in C}\sum_{l\in L} hc_c\cdot S_{c,l}$). The METRIC type methods implicitly find the best way of dividing the allowed unavailability over the components, using a so-called system approach.

4.2 Iterative approach

Our iterative approach uses the same two building blocks as the sequential approach does. However, we use the annual spare parts holding costs that result to adapt the LORA inputs and perform a second iteration of LORA and spare parts stocking. The goal of the feedback loop is to incorporate an estimate of the spare parts holding costs that result from a LORA decision in the LORA problem, thus aiming to find a LORA solution that leads to low total costs. In terms of our mathematical model formulation in Section 3.3.4, we first solve the model without the last term in the objective function, and without Constraint 6. Then, using the repair decisions ($\mathscr{X}$) as an input, we use VARI-METRIC to determine the amount of spare parts to stock ($\mathscr{S}$) in the complete network, such that Constraint 6 is satisfied and the spare parts holding costs ($\sum_{c\in C}\sum_{l\in L} hc_c\cdot S_{c,l}$) are minimized. In our feedback loop, we then add the calculated spare parts holding costs to the (variable or fixed) LORA costs, so that these are taken into account in the LORA problem in the next iteration (the model without the last term in the objective function, and without Constraint 6). Below, we discuss the basic idea of our algorithm and the feedback loop. A formal description of our algorithm, including the stopping criterion, is given in Section 5.

4.3 Feedback mechanism

After one iteration, we have a LORA solution, consisting of repair/discard decisions for each component, ignoring spare parts holding costs, and the spare parts stocking solution, consisting of a stock allocation in the complete network and resulting spare parts holding costs for each component. The key idea is that, as an approximation, we may decompose the spare parts holding costs into spare parts holding costs per component, so that for each component c the spare parts holding costs are $\sum_{l\in L} hc_c\cdot S_{c,l}$. In this way, we

assume that the spare parts holding costs that result from a repair/discard decision are independent of the decisions taken for the other components. As discussed above, this assumption is violated.

We feed back the spare parts holding costs of each component in the entire network to the LORA problem; the costs are added to the costs of the repair/discard option that was chosen for that component (the technical specification can be found in Section 5). Most probably, this repair/discard option will now be expensive, compared with the other possible repair/discard options, so that another option is chosen in the second iteration. A second spare parts stocking analysis is performed and the new spare parts holding costs are decomposed and added to the repair/discard decision taken for that component in the second iteration. By performing a couple of iterations, we will gradually find spare parts holding costs for more repair/discard decisions. We expect that after a couple of iterations, the LORA will recommend an option that leads to low total life cycle costs: LORA costs (excluding the added spare parts holding costs), plus spare parts holding costs resulting from the spare parts stocking analysis.

Before we move to the technical description, we point out that we encounter two issues. First, our algorithm does not converge monotonically. Instead, the total costs may both increase and decrease during the iterations, and even cycling between two (or more) solutions is possible. This mean that we have to define a stopping criterion carefully, see Section 5.2. Second, we may find different estimates for the spare parts holding costs related to a specific repair/discard decision for a certain component, since these costs are influenced by the decisions taken for the other components. In our basic algorithm, we always use the newest estimate in the next iteration. However, another option is to use a weighted average of the two estimates, see Section 5.3.

5 Algorithm

We now explain the algorithm in more detail. We distinguish four parts in our algorithm:

- the LORA building block: the minimum cost flow formulation of Basten et al. (2008);
- the spare parts stocking building block: VARI-METRIC;
- the feedback loop: discussed in Section 5.1;
- the stopping criterion: discussed in Section 5.2.

We further propose two variants of the basic algorithm in Section 5.3.

5.1 Feedback mechanism

In principle, we can include the spare parts holding costs in both cost types that we use in the LORA inputs; we choose to include them in the variable costs. We add to the original variable costs $vc_{c,e,d} \mid d \in \{\text{repair}, \text{discard}\}$ the stored spare parts holding costs $vc^{\text{s}}_{c,e,d,j}$ after iteration j (input for iteration $j+1$). In the input of the first iteration, all $vc^{\text{s}}_{c,e,d,0} = 0$. At the end of each iteration j, we set for each tuple $(c,e,d,) \mid d \in \{\text{repair}, \text{discard}\}$ for which $X_{c,e,d} > 0$ in iteration j: $vc^{\text{s}}_{c,e,d,j} = \frac{\sum_{l \in L} S_{c,l} \cdot hc_c}{\lambda_c}$. For all other repair and discard decisions, we set $vc^{\text{s}}_{c,e,d,j} = vc^{\text{s}}_{c,e,d,j-1}$.

We explain the feedback mechanism using an example. We consider a radar system that consists of two components (A and B). The radar system is installed at two ships, which are supported by a depot. So, the example considers a two-echelon, single-indenture problem instance. Both component A and B require a unique resource in order to enable repair. The fixed annual costs of these resources are €10,000 and €25,000, respectively. Furthermore:

- the annual failure rate is 1 per component per ship;
- variable move costs are €0;
- variable repair costs are €5,000;
- variable discard costs are €15,000.

In the first iteration, there are no spare parts holding costs in the LORA problem. Therefore, the repair/discard options with the lowest LORA costs are chosen. For component A this is 'repair at depot',

	LORA costs		Spare parts holding costs ($vc^s_{c,e,d,j}$) at the end of					
			iteration 1		iteration 2		iteration 3	
	Component		Component		Component		Component	
Decision	A	B	A	B	A	B	A	B
Repair at ship	30	60	0	0	***4***	0	4	0
Repair at depot	20	35	***12***	0	12	***15***	12	***20***
Discard	30	30	0	***22***	0	22	***20***	22
Total costs (LORA and spares)			84		84		105	

Table 1: Costs in the LORA problem (x €1,000)

since that leads to annual repair costs of €10,000 (2 failures, 1 at each ship) and a resource at the central depot; for component B this is 'discard', since that leads to annual discard costs of €30,000 and no resource costs. The annual LORA costs for each repair/discard option can be found in the second and third columns of Table 1. Next, we solve the spare parts stocking problem, and we find that spares for component A should be stocked at both the ships and the depot, leading to annual spare parts holding costs of €12,000 for component A and €22,000 for component B. The fourth and fifth columns in Table 1 show the spare parts holding costs in our LORA input for the second iteration. The costs that are changed are bold and italic.

In the second iteration, we solve the LORA with the modified inputs. The LORA will choose for component A one of the two options 'repair at ship', or 'discard', since both options lead to total costs of €30,000, whereas repair at depot leads to total costs of €20,000 + €12,000. We assume that 'repair at ship' is chosen. For component B, 'repair at depot' is the most cost effective option. After solving the spare parts stocking problem, we find that the spare parts holding costs are €4,000 and €15,000 for the two components, respectively. We adapt the LORA inputs accordingly.

In the third iteration, it is decided to discard component A. For component B, it turns out that 'repair at ship' is not an interesting option, even if this leads to zero spare parts holding costs. As a result, component B is repaired at depot. Executing the spare parts stocking analysis leads to annual spare parts holding costs of €20,000 for both components A and B. Notice that we do not modify the LORA decision for component B, but the spare parts holding costs for component B are changed, due to a change in the decision for component A. We simply replace the old stored costs by the newly calculated costs (in the basic algorithm). Notice that the costs of €20,000 mean that the option to repair component B at depot has become so expensive, that it is not chosen anymore in later iterations, although the costs may be lower if combined with another decision for component A (for example, repairing both components at depot). The solution in the next iterations will be to repair component A at depot and to discard component B.

The key drawback of our approach is the risk that our feedback mechanism results in storing spare parts holding costs that are too high, so that we will not select that option in the LORA anymore. As a result, we may end up with a non-optimal solution. We try to reduce this problem by developing variants of our algorithm in Section 5.3. Although we cannot give an indication of the quality of our algorithm compared with the optimal solution, we do show in the numerical experiments in Section 6 what we gain compared with the sequential approach.

5.2 Stopping criterion

As mentioned in Section 4, the stopping criterion for our algorithm is not trivial because the objective function does not decrease monotonically during the iterations and may even converge to a value above

the best value encountered during the iterations. Stopping when the total costs in two consecutive solutions are almost equal does therefore not work, the costs being the total of the LORA costs (without the added spare parts holding costs), plus the spare parts holding costs resulting from the spare parts stocking analysis. We have to take into account two events that may occur:

1. Two solutions are chosen alternately ('cycling'), so that the algorithm will not find better solutions anymore, and it should terminate.
2. The costs in two consecutive iterations are almost equal, but spare parts holding costs in the LORA problem are still being changed (values $vc^{\mathrm{s}}_{c,e,d,j}$, which are not part of the objective function). If the costs are still being changed, the LORA might still find a better solution during a later iteration. This may happen in the first few iterations, when the algorithm is still 'exploring' all the options.

We can cope with the occurence of both events using a stopping criterion consisting of two conditions that should both be satisfied:

1. After each iteration, we store the solution if it is better than the best we have found thus far. We may stop our algorithm if we did not find a better solution during j_1 iterations.
2. In each iteration, we update the spare parts holding costs estimates that we add to the variable costs in the LORA inputs. We assume that these estimates are accurate enough if the total spares parts holding costs that is part of the LORA solution ($\sum_{c\in C}\sum_{e\in E}\sum_{d\in D} vc^{\mathrm{s}}_{c,e,d,j}\cdot\lambda_c\cdot X_{c,e,d}$) deviates less than $p\%$ from the total spare parts holding costs that is calculated in the spare parts stocking problem after the LORA has been solved ($\sum_{c\in C}\sum_{l\in L} hc_c\cdot S_{c,l}$). We may stop calculations j_2 iterations afterwards.

The choice of the number of iterations j_1 and j_2 and the percentage difference p is more or less arbitrary. We choose $j_1 = j_2 = 10$ and $p = 1\%$ in our experiments. We performed tests on 80 of the most difficult problem instances to see whether results change if $j_1 = j_2 = 100$ and $p = 0.1\%$, but for none of the problem instances they did. Therefore, we conclude that our stopping criterion serves its purpose in the sense that we do not stop too early (it may be possible to stop even earlier, but this is less important given the speed of the algorithm, see Section 6).

5.3 Variants

As mentioned in Section 5.1, the spare parts holding costs for a certain repair/discard option may be very high in a single iteration due to the decisions that have been taken for the other components. As a result, that option is not chosen in later iterations anymore, although it may be an interesting option if alternative decisions are taken for the other components. We discuss two variants of our algorithm to cope with this issue in Sections 5.3.1 and 5.3.2, respectively. We also performed tests in which we tried to improve the starting solution of our algorithm by making an initial estimate of the stored spare parts holding costs for all repair/discard options. However, this did not improve our algorithm.

5.3.1 Using a weighted average to update stored spare parts holding costs

If our algorithm finds spare parts holding costs for a certain repair/discard option, we add these to the inputs of the LORA problem, by simply replacing the old value (possibly being zero) with the new value. If the costs are very high in one iteration, the spare parts holding costs that we store are very high as well and the corresponding repair/discard option may be excluded in the LORA in any later iteration. Therefore, we propose to take a weighted average of the old and new value. If we get a new spare part holding costs estimate for option d at echelon e for component c at the end of iteration j, we store the value $vc^{\mathrm{s}}_{c,e,d,j} = \alpha\cdot vc^{\mathrm{est.}}_{c,e,d,j} + (1-\alpha)vc^{\mathrm{s}}_{c,e,d,j-1}$, with:

- $vc^{\mathrm{s}}_{c,e,d,j}$ being the spare parts holding costs stored in the database for option d at echelon level e for component c at the end of iteration j ($vc^{\mathrm{s}}_{c,e,d,0} = 0$);
- α being the weight; and

- $vc^{\text{est.}}_{c,e,d,j}$ being the estimate for the spare parts holding costs for option d at echelon level e for component c that results from the spare parts stocking analysis at the end of iteration j.

If we select the same option in a number of consecutive iterations, we gradually update the spare parts holding costs to the value that we find repeatedly. Notice that we only adapt spare parts holding costs for the options that are chosen in the last iteration. Two drawbacks of using a weighted average are that the number of required iterations will grow and that there is no guarantee that the solution that is found is at least as good as the solution that is found using the basic algorithm.

5.3.2 Decreasing stored spare parts holding costs

Instead of preventing storing spare parts holding costs in the LORA inputs that are too high, we may also decrease those costs after they have become too high. However, we do not know which costs are too high, when to lower them, and with how much to lower them. There are various options. For example:

- Decrease the spare parts holding costs for repair/discard options that have not been chosen in the last j_3 iterations (for example, $j_3 = 10$).
- Decrease the spare parts holding costs for randomly picked repair/discard options (x_1% of the total number of repair/discard options, with, for example, $x_1 = 25$).
- Decrease the x_2% highest stored spare parts holding costs (for example, $x_2 = 25$).

Since each of the options has pros and cons, we choose a relatively simple approach (we also tested a more sophisticated approach, but the change in results was minor), inspired by the variable neighborhood search (see, e.g., Mladenović and Hansen, 1997): we choose to lower the costs at the moment that the basic algorithm reaches the stopping criterion. In this way, we try to get the algorithm out of a (local) optimum, in order to find a better (local) optimum. We do this by restoring the stored spare parts holding costs that we had in the best iteration (the lowest total costs) and decreasing these costs by a certain percentage p_1 for all repair/discard options, except for those that were picked in the LORA solution in the best iteration, since those values are up-to-date. We also reset our stopping condition.

If our stopping condition is reached again without finding a better solution, we restore the best solution again, and decrease all stored spare parts holding costs with a percentage $p_2 > p_1$ (again, not those that were set in the best iteration), et cetera. If we do find a better solution, we store that solution, and when the stopping condition is reached, we lower the restored spare parts holding costs with p_1, et cetera. Obviously, the number of iterations that is required will grow. However, the solution that is found is at least as good as the solution that is found using the basic algorithm, since that solution is found the first time the stopping criterion was reached.

6 Computational experiments

We design an extensive numerical experiment that we present in Section 6.1. In Section 6.2, we discuss the results of our tests by answering the questions:

1. What cost reduction can be achieved by performing the LORA and spare parts stocking analysis iteratively, compared with performing them sequentially?
2. Which variant of our iterative method gives the best results?
3. How do the repair strategies change if we use the iterative approach instead of the sequential approach?
4. For which type of problem instances does the iterative approach yield most cost reductions?

6.1 Design

In our experimental design, we restrict ourselves to problem instances that are symmetrical in the network structure, the cost factors, the demand rates, and the throughput times. Due to the symmetrical

structure, we decide to make the spare parts decisions symmetrical as well: we stock the same number of spare parts at each location at one echelon level. This approach is optimal for symmetrical networks, except that the overshoot increases (the achieved availability may be higher than the target availability).

We use three sets of problem instances. Each has its own focus, which we explain in Section 6.1.2. We use a full factorial design, in that we test each possible combination of parameter settings in each set. If we give a range for a parameter, we randomly draw values from the given range. These values are the same for all settings of the other parameters. For each combination of parameter settings, we generate ten problem instances, in order to obtain a variety of problem instances. Since each parameter setting has a default value (or range) that is the same in the three test sets, there are ten problem instances that are part of each of the three sets. The parameter settings for these ten problem instances are used in Section 6.1.1, where we explain how we generate problem instances. Details can be found in Appendix 8.

6.1.1 Default scenario

The repair network consists of three echelon levels, with a central depot, two intermediate depots, and ten operating sites. The lead times in the network are as follows.

- the discard time, so the time it takes to receive a new component at the central depot, is in the range [1/10 – 1/2];
- the replenishment lead time from one echelon to the next is in the range [2/52 – 4/52];
- the repair time is in the range [2/52 – 4/52].

The product structure consists of three indenture levels, with 50 LRUs, 100 SRUs, and 200 parts. The failure rate of a component is equal to the failure rate of its subcomponents. Since the failure rate of a part is in the range $[0.01/2^2 – 0.25/2^2]$, the failure rate of each LRU is close to the range $[0.01 – -0.25]$ (see for details Appendix 8). The net component price is in the range [1,000 – 10,000]. Using these prices, we calculate the variable costs as follows:

- repair costs as a fraction of the net component price are in the range [25% – 75%];
- we recursively add the costs of each subcomponent to its parent to get the gross component price of the parent;
- the discard costs as a fraction of the gross component price, are in the range [75% – 125%];
- the move costs as a fraction of the gross component price are 1%;
- the annual costs of holding one spare part of a component are 20% of the gross component price.

There are ten resources and their annual costs are in the range [10,000 – 100,000]. 50% of the resources is required by one component only, the other 50% is required by 2 to 6 components. We distinguish 4 'component types', for example electronic versus mechanical components. Each resource and each LRU family (an LRU including all its subcomponents at any indenture level) is randomly assigned to one of the component types. The result is that resources of one component type do not interact with resources of another component type, which is realistic in practice.

6.1.2 Three test sets

The problem instances are divided into three sets, as mentioned above. For each of these sets, we explain the focus and we give the values for each of the parameters that are varied:

1. We vary the problem size, the spare parts holding costs and the lead times: # echelons: 2 & 3, # indentures: 2 & 3, # LRUs: 50 & 100, annual holding costs: [20% – 20%] & [20% – 40%], discard time: [1/10 – 1/2] & [1/4 – 1/2], repair time: [0.5/52 – 4/52] & [2/52 – 4/52], and move time: [0.5/52 – 4/52] & [2/52 – 4/52] (1,280 instances).
2. We vary the attractiveness of acquiring resources by changing the annual number of failures and the costs of resources and components (and thereby the variable repair, discard, and move costs): annual demand of LRU: [0.01 – 0.1], [0.01 – 0.25], [0.01 – 0.5] & [0.01 – 1], net component

α	Number of iterations	Cost reduction compared with sequential average	minimum	maximum	Cost reduction compared with basic iterative average	minimum	maximum
—	17	2.73%	0.00%	34.69%	—	—	—
0.9	22	2.88%	0.00%	35.49%	0.15%	-2.26%	2.79%
0.8	25	2.90%	0.00%	35.57%	0.18%	-3.55%	2.54%
0.7	29	2.92%	0.00%	34.94%	0.20%	-2.70%	2.51%
0.6	33	2.91%	0.00%	34.95%	0.19%	-2.34%	3.07%
0.5	39	2.94%	0.00%	34.94%	0.22%	-3.61%	2.77%

Table 2: Varying the weighted average α

price: [1,000 – 10,000] & [1,000 – 100,000], annual resource cost: [10,00 – 100,000] & [10,00 – 500,000] (160 instances).

3. We vary the component-resource relations in various ways: # Component types: 3 & 4, % resources used by 1 component: 0% & 50%, # components per resources: 2 to 3 & 2 to 6 (80 instances).

6.2 Results

We subsequently address the four questions that we posed at the start of Section 6.

6.2.1 What cost reduction can be achieved compared with the sequential approach?

Our basic algorithm yields a cost reduction over the 1,280 problem instances in test set 1 of 2.73% on average compared with using the sequential method (almost 35% at maximum). As we will see at the end of Section 6.2.2, the best variant of our algorithm yields an average cost reduction on this test set of 3.04%. Over all three test sets this cost reduction is 3.17% on average and over 35% at maximum. In almost 9% of the problem instances, we achieve a cost reduction of over 10%. Since a cost reduction of a few percent may be worth hundreds of thousands of euros over the life cycle of a capital good, it is clearly risky to rely on the sequential approach.

6.2.2 Which variant of our iterative method gives the best results?

To test which variant of our algorithm performs best, we use test set 1 only. Table 2 shows that using a weighted average to update the stored spare parts holding costs improves the results compared with using our basic algorithm (first row). With a minor bump for $\alpha = 0.6$, the average cost reduction keeps increasing with a decreasing α. However, the number of iterations keeps increasing as well, but it does certainly not explode. Over all tests, we found a maximum of 66 iterations or 2.5 minutes (both with $\alpha = 0.5$). As a result, we recommend to take α as low as possible for problem instances in practice, but for our further tests, we take $\alpha = 0.7$, since that gives a good combination of cost reduction and number of iterations.

Table 3 presents the results of decreasing the stored spare parts holding costs with a certain percentage when the stopping criterion is met. We see that it does reduce the costs with just over 0.2% on average compared with our basic algorithm. We also see that the performance of both sets of decrease factors does not differ much, and that for both sets, the number of iterations increases, as expected; this leads to an average optimization time of about one and a half minute. In contrast to using a weighted average, using decrease factors never yields a solution that is worse than the solution of the basic algorithm (see also Section 5.3.2). Since both settings yield approximately the same results, we propose to

Decrease factors[a]	Number of iterations	Cost reduction compared with sequential			Cost reduction compared with basic iterative		
		average	minimum	maximum	average	minimum	maximum
—	17	2.73%	0.00%	34.69%	—	—	—
5% – 10% – 20%	66	2.93%	0.00%	34.69%	0.21%	0.00%	2.77%
10% – 25% – 50%	70	2.94%	0.00%	35.13%	0.22%	0.00%	2.77%

Table 3: Varying the decrease factors

[a] $p_1 - p_2 - p_3$ means that if the stopping condition is met the first time, values are decreased with p_1, if no better solution is found, values are decreased with p_2, and if still no better solution is found, values are decreased with p_3.

Echelon level	Sequential	Iterative	Difference
Central depot	0.00	1.87	+1.87
Intermediate depots	0.00	0.89	+0.89
Operating sites	64.63	61.66	-2.97
Total	64.63	64.42	-0.21

Table 4: # annually repaired LRUs

use the decrease factors of 5% – 10% – 20%, since that leads to the least number of iterations. This may especially be important if the decrease factors are combined with using a weighted average.

Based on the results for each of the variants individually, we perform tests using a weighted average $\alpha = 0.7$ and the decrease factors 5% – 10% – 20%. The average cost reduction compared with using the sequential method is 3.04% in test set 1, which means that the additional cost reduction over using the basic iterative algorithm is 0.32% on average (compared with 0.20% for using only $\alpha = 0.7$ and 0.21% for using only the decrease factors 5% – 10% – 20%). The cost reduction compared with the basic iterative algorithm is -2.70% at minimum (a cost increase), and 2.73% at maximum. This suggests that in practice, it may be interesting to use a number of settings when solving a problem instance and to choose the best solution. Over all three test sets, the average cost reduction compared with the sequential approach is 3.17%, and it is over 35% at maximum. The maximum number of iterations that is required is 222, and the maximum running time is just over ten minutes.

6.2.3 How do the repair strategies change if we use the iterative approach?

We perform the tests in this and next section using the best combination of variants, which we found at the end of the previous section The iterative procedure places more resources in the network than the sequential approach does. Still, the number of resources is small: 1.35 (1.18 at central depot) and 1.11 (1.10 at central depot) on average, respectively. As a result, slightly more repairs and less discards are performed in the iterative solution. In addition, the iterative algorithm recommends to perform some more repairs at the higher echelon levels, instead of at the operating sites for components that do not require resources. To clearly show this effect, we focus on the 470 problem instances in test set 1 for which no resources are located in the results of either the sequential or the iterative algorithm. The solution is simple for the sequential algorithm: repair all components that require no resource at the operating site, thus avoiding move costs, and discard all other components. In the result of the iterative algorithm, some repairs are performed at a more central location (and some more components are discarded), see Table 4. The reason is that if repairs are performed at a higher echelon level, pooling effects for the spare parts can sometimes be used: instead of stocking one spare part at each operating site, leading to ten spare parts in total, maybe only two spare parts are required at each of the intermediate depots, leading to four spare parts in total.

Parameter	Setting	Cost reduction compared with sequential Average	Maximum
# LRUs	50	4.80%	35.02%
	100	1.28%	13.60%
Move lead time	[0.5/52 – 4/52]	4.91%	35.02%
	[2/52 – 4/52]	1.17%	6.08%
Demand per LRU	[0.01 – 0.1]	11.61%	17.00%
	[0.01 – 0.25]	2.81%	5.19%
	[0.01 – 0.5]	1.43%	4.07%
	[0.01 – 1]	3.31%	7.91%

Table 5: Cost reduction for the interesting parameter settings

6.2.4 For which type of problem instances does the iterative approach yield most cost reductions?

Table 5 gives the cost reductions that the iterative algorithm achieves compared with the sequential algorithm for the parameter settings that have a serious impact on the cost reductions that we achieve in test sets 1 and 2, respectively. We discuss these results below; the other parameters that we varied in test sets 1, 2, and 3 do not have a big impact on the achieved cost reductions.

The influence of the number of LRUs on the cost reduction that can be achieved relates to the total unavailability of 5% (=100%-95%) that is allowed for the complete product. If we want to achieve the same availability for a product with 50 LRUs as for a product with 100 LRUs, then the unavailability that is allowed per LRU is about twice as low in the case of 100 LRUs. If the unavailability that is allowed per LRU is so low, then using pooling effects with the spare parts is often not possible: if there are no spares at the operating sites, the achieved availability is already too low. To verify that this explains the results we performed a test for all problem instances with 50 LRUs, aiming at an availability of 97.5%. In that case the cost reduction that can be achieved decreases to 2.03%, which is still higher than 1.28% (100 LRUs, 95% target availability), but much lower than 4.80% (50 LRUs, 95% target availability).

Intuitively, long lead times lead to high spare parts holding costs, and therefore, much can be gained using an iterative procedure. However, this is not what the results show: if the move lead time is relatively long, using pooling effects is not possible; if spares are not located at the operating site, there is always a long downtime if a failure occurs, since it takes a while before the spare part arrives. This means that with increasing lead times, the cost reduction that can be achieved decreases.

The influence of the demand per LRU on the cost reduction that can be achieved may be explained as follows. If the demand per LRU is very low ([0.01 – 0.1]), pooling effects can be used more effectively than if demands are higher. The costs for spare parts at the central depot increase with 69% compared with the sequential solution in the case of very low demands, whereas it decreases with 11% on average for the higher demands. The reason is that if demands are very low, the availability will be high without storing many spares. For higher demands, storing at least one spare part at a low echelon level may be required to achieve the target availability, so that the iterative algorithm is not able to lower the costs.

7 Case study at Thales Nederland

We performed a case study at Thales Nederland, which we describe in Section 7.2. The goal of this study is to find out which cost reduction we may obtain in practice and which advantages and drawbacks of our new approach we can identify for application in practice. In Section 7.1, we discuss the current way of working at Thales Nederland to solve the LORA and spare parts stocking problem, and the difficulties

this leads to. In Section 7.3, we compare the results of using our iterative algorithm with those of using the sequential algorithm and those of the logistic engineers at Thales Nederland.

7.1 Current way of working

Logistic engineers at Thales Nederland first conduct a so-called *non-economic LORA*. The goal of this non-economic LORA is to exclude non-realistic repair or discard options and to simplify the problem. Questions that are posed are, for example:

- Is the component prone to failure?
- Does the customer prescribe the maintenance policy for the component? If so, this policy is followed.
- Does the value of the component exceed a certain threshold? If not, it can be discarded by default, since it is not worth repairing.

The result is that for some components, repair/discard options may be excluded. If one option only remains, no decision needs to be taken in the LORA problem for that component, but the component is still taken into account in the spare parts stocking problem, because of its influence on the availability. Furthermore, only part of the resources are included in the LORA, mainly the expensive ones.

After the input data has been structured and filtered, the logistic engineer finds a first solution. He uses decisions made for previous products, his experience, and spreadsheet calculations. Then, he uses a spare parts stocking tool (INVENTRI, based on VARI-METRIC and the work of Rustenburg, 2000) to stock spare parts. Analyzing the results, he finds components for which spare parts holding costs are very high. If he thinks that this might help, he changes the LORA decision for these components, calculates the new LORA costs and solves a spare parts stocking problem. So in fact, the logistic engineer tries to perform some manual iterations, which has as its drawbacks that this approach is:

- Time consuming, since an analysis takes up to a few days after all data has been acquired.
- Non-reproducable, because of the judgmental feedback loop.
- Error sensitive.

7.2 Case: a sensor system

We consider a sensor system (combined radar and electro-optical surveillance system) manufactured by Thales Nederland. Although the product structure consists of six indenture levels, we consider only three indenture levels, as a result of the non-economic LORA. For the same reason, the product structure consists of over 1,500 components, but only slightly more than 200 turn out to be relevant, of which 40% are LRUs. For about one third of the components, only one repair/discard option remains, and for an additional one third, the repair/discard options that can be chosen are restricted. The repair network consists of twelve ships, attached to two intermediate depots, a central depot and the OEM. There are 54 resources, 34 of which are 'adapters'. These adapters are used in concurrence with other test equipment.

The costs of the various components can be up to one million euros, and the costs of the various resources can be up to a couple of million euros. These costs are not used directly. Instead, there are three types of costs in the joint problem of LORA and spare parts stocking: variable costs per repair or discard action, fixed annual costs for locating resources, and annual spare parts holding costs. For each type of costs we give the most important cost factors that we include (another overview of the cost factors to include can be found in Saranga and Dinesh Kumar, 2006):

- Variable repair costs (customer's network): working hours (e.g., locating failure, exchanging subcomponents, and performing direct repair), variable costs for using resources (e.g., energy consumption and wear), and usage of additional parts (e.g., bulk items such as screws and wires).
- Variable repair costs (OEM and outsourced in general): listed repair price.
- Variable discard costs: procurement price for the component that replaces the discarded component and disposal costs or a residual value of the discarded component.

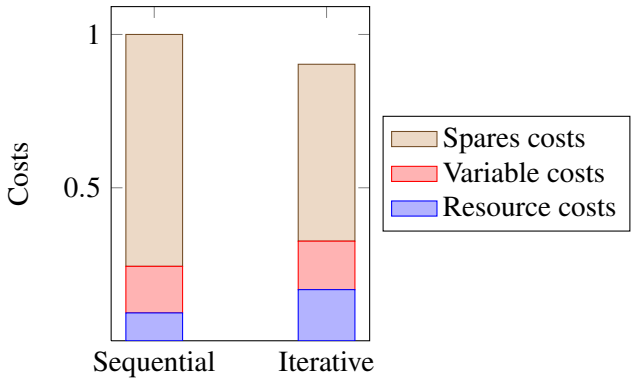

Figure 3: Costs for Thales case (normalised)

- Variable move costs: transportation costs and handling and administrative costs.
- Fixed resource costs: depreciation costs, costs of capital, a risk factor (e.g., insurance against damage and theft), fixed operating costs (e.g., a location to operate the equipment), and maintenance costs of the resource. Resources may have a residual value after their economic lifetime.
- Spare parts holding costs: costs of capital, a risk factor, and storage costs. Spares may have a residual value after the lifetime of the product.

7.3 Results

We solved the problem instance using the best settings for the iterative algorithm (see Section 6.2.2). This takes less than one and a half minute (20 iterations) and Figure 3 shows the results. More resources are installed and more repairs are performed in the customer's network and in total. This leads to higher resource costs and higher variable costs, but also to much lower spare parts holding costs. Since spare parts tend to be very expensive in the defence industry, the overall effect is a cost reduction of 9.7% compared with the sequential method, which is worth millions over the life time of twelve sensor systems. The cost reduction is achieved by:

- Installing two resources at the depot that are not installed in the sequential solution.
- Installing one resource at both intermediate depots instead of one at the central depot.
- Installing one resource at all ships instead of one at each of the two intermediate depots.

The other resources are installed at the same echelon level in both solutions. The logistic engineer at Thales Nederland achieves about a quarter of the cost reduction that we achieve using the iterative algorithm. He locates resources at a more central location than our iterative algorithm does, but at a more decentralized location than the sequential algorithm does.

8 Conclusions and recommendations for further research

We presented an algorithm that can be used to perform the LORA and spare parts stocking analysis in an iterative way for multi-indenture, multi-echelon problem instances with very mild restrictions on the resource-component relations. Our conclusions are as follows:

- We have shown that we can solve real life problems by solving a case study at Thales Nederland. We can solve the case study in about one and a half minute and achieve a cost reduction of almost 10% compared with performing the LORA and spare part stocking analysis sequentially. We also generated problem instances, consisting of up to 700 components. We can solve these instances in just over ten minutes at maximum and we achieve a cost reduction of over 3% on average, with a maximum of over 35%.

- The best variant of our algorithm combines using a weighted average when updating spare parts holding costs, with $\alpha = 0.7$, and decreasing costs when the stopping condition is reached, using decrease factors 5% – 10% – 20%.
- The major drawback of our approach is that we do not know whether the solutions that we find are close to optimal.

Interesting further research is as follows:

- Developing an algorithm that is guaranteed close to optimal, for example, by giving an efficient frontier of optimal combinations of availability and costs, as in the METRIC type methods. This algorithm may be used by itself, or it could be used to check the quality and robustness of our iterative algorithm. At the moment of writing, we are working on such an algorithm.
- Improving our iterative algorithm. For example, by using a number of variants of the algorithm to solve one problem instances, and then choosing the best solution that is found by any of the variants, or by developing new variants, possibly improving the starting solution.
- Extending our model to include, for example, non-symmetrical LORA decisions, repair probabilities, or capacitated resources. The key problem with such extensions is the feedback mechanism. In our current algorithm, we make one approximation in the feedback mechanism: we decompose the spare parts holding costs into spare parts holding costs per component. The abovementioned extensions require more approximations. Extensions that affect only one building block, such as certain flexibility options in the spare parts stocking analysis, may be included easily.

Acknowledgments

The authors are thankful to Martijn Smit for his work on the case study. The authors gratefully acknowledge the support of the Innovation-Oriented Research Programme 'Integrated Product Creation and Realization' (IOP IPCR) of the Netherlands Ministry of Economic Affairs and the support of TRANSUMO.

References

Alfredsson, P. (1997). Optimization of multi-echelon repairable item inventory systems with simultaneous location of repair facilities. *European Journal of Operational Research*, 99:584–595.

Barros, L. L. (1998). The optimization of repair decisions using life-cycle cost parameters. *IMA Journal of Mathematics Applied in Business & Industry*, 9:403–413.

Barros, L. L. and Riley, M. (2001). A combinatorial approach to level of repair analysis. *European Journal of Operational Research*, 129:242–251.

Basten, R. J. I., Schutten, J. M. J., and Van der Heijden, M. C. (2009). An efficient model formulation for level of repair analysis. *Annals of Operations Research*. (In press).

Basten, R. J. I., Van der Heijden, M. C., and Schutten, J. M. J. (2008). A minimum cost flow model for level of repair analysis. BETA working paper 254.

Brick, E. S. and Uchoa, E. (2009). A facility location and installation or resources model for level of repair analysis. *European Journal of Operational Research*, 192(2):479–486.

Ferrin, B. G. and Plank, R. E. (2002). Total cost of ownership models: An exploratory study. *Journal of Supply Chain Management*, 38(3):18–29.

Mladenović, N. and Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, 24(11):1097–1100.

Muckstadt, J. A. (2005). *Analysis and Algorithms for Service Parts Supply Chains*. Springer, New York (NY).

Rustenburg, W. D. (2000). *A System Approach to Budget-Constrained Spare Parts*. PhD thesis, BETA research school, Eindhoven (The Netherlands).

Saranga, H. and Dinesh Kumar, U. (2006). Optimization of aircraft maintenance/support infrastructure using genetic algorithms — level of repair analysis. *Annals of Operations Research*, 143:91–106.

Sherbrooke, C. C. (1968). METRIC: A multi-echelon technique for recoverable item control. *Operations Research*, 16(1):122–141.

Sherbrooke, C. C. (2004). *Optimal inventory modelling of systems. Multi-echelon techniques*. Kluwer, Dordrecht (The Netherlands), second edition.

Appendix A: problem instances generator

We explain how we generate the problem instances that we use in our experiments. For each parameter that we use to generate these instances, we use the default setting (value or range) in the text. Tables 6 and 7 then summarize all parameters with their default setting and show the alternative setting that we use and the test set in which we use that alternative setting. The problem instances are divided into three test sets, each with its own focus:

1. The sizes of the product structures and network structures are varied, together with lead times and holding costs.
2. Costs of components and resources and annual demands are varied.
3. Component-resource relations are varied.

We use a three-echelon repair network that is completely symmetrical in the cost factors, the demand rates, and the throughput times. It consists of a central depot, two intermediate depots, and ten operating sites. The three-indenture product structure consists of 50 LRUs, 100 SRUs, and 200 parts (so two subcomponents per component on average). Since each subcomponent is randomly assigned to one of the components, in general, the number of subcomponents per component differs for the various components. The annual demand for a component is equal to the annual demand of its subcomponents. We achieve this by drawing the annual demand of each part from a uniform distribution on the interval $[0.01/(\#\text{subcomp. per comp.})^2 - 0.25/(\#\text{subcomp. per comp.})^2]$, and recursively calculating the annual demand of the SRUs and LRUs. The demand for SRUs or LRUs without subcomponents is drawn from the same interval as the demand for parts.

For each component, we draw a net price from a shifted exponential distribution with shift factor 1,000 and rate parameter $7/(100{,}000 - 1{,}000)$. As a result, we do not have components with a price below 1,000, since they are typically discarded by default. Furthermore, there are considerably more cheap components than expensive ones. On average 1‰ of the components get a value larger than 100,000, but we draw a new price for these components to avoid odd problem instances. Using the calculated net prices, we calculate the variable costs as follows:

- Repair costs as a fraction of the net component price are drawn from a uniform distribution on the interval [0.25 – 0.75].
- Next, we recursively add the costs of each subcomponent to its parent to get the gross component price for the parent.

- The discard costs as a fraction of the gross component price are drawn from a uniform distribution on the interval [0.75 – 1.25]. These costs include the costs of purchasing a new component and the disposal costs or residual value of the defective component.
- The move costs as a fraction of the gross component price are 1%.
- The annual costs of holding one spare part of a component are 20% of the gross component price.

For each component, we draw a repair time from a uniform distribution on the interval [2/52 – 4/52]. The discard time, the time it takes to order a component and receive it at the central depot, is drawn from a uniform distribution on the interval [1/10 – 1/2]. Both the discard and the repair times vary over the components, but are the same at all echelon levels. The replenishment lead time from one echelon to the next is drawn from a uniform distribution on the interval [2/52 – 4/52]. This value varies over the echelon levels, but is the same for all components.

There are ten resources and their annual costs are drawn from a shifted exponential distribution with shift factor 10,000 and rate parameter $7/(100,000 - 10,000)$. 50% of the resources will be used by one component only, the other 50% will be used by 2 to 6 components. We distinguish 4 'component types', for example electronic versus mechanical components. Each resource and each LRU family is randomly assigned to one of the component types. The result is that resources of one component type do not interact with resources of another component type.

Table 6 gives the values for the parameters that do not vary over a range and Table 7 gives the values for the parameters that do vary. For each combination of parameters, we generate 10 problem instances to decrease the risk of basing conclusions on one odd problem instance. This means that set 1 consists of 1,280 problem instances, set 2 consists of 160 problem instances, and set 3 consists of 80 problem instances. Each test set contains the same ten problem instances that are generated using the default setting for each parameter.

Parameter	Default value	Test set	Additional value(s)
# Echelon levels	3	1	2
# Central depots	1	—	—
# Intermediate depots	2	—	—
# Operating sites[a]	10	—	—
# Indenture levels	3	1	2
# LRUs	50	1	100
# Subcomponents per parent component	2	—	—
# Resources	10	—	—
# Component types	4	3	3
% Resources used by 1 component	50%	3	0%

Table 6: Fixed values

[a] In the tests with 2 echelon levels (test set 1), the number of operating sites is 5. In this way, there are always five operating sites attached to the supplying location at the next higher echelon.

Parameter	Default range	Test set	Additional value(s)
Annual demand of LRU	0.01 – 0.25	2	0.01 – 0.1 & 0.01 – 0.5 & 0.01 – 1
Net cost of component	1,000 – 10,000	2	1,000 – 100,000
Discard costs	75% – 125%	—	—
Repair costs	25% – 75%	—	—
Move costs	1% – 1%	—	—
Annual holding costs	20% – 20%	1	20% – 40 %
Annual cost of resource	10,000 – 100,000	2	10,000 – 500,000
Discard time (in years)	1/10 – 1/2	1	1/4 – 1/2
Repair time (in years)	0.5/52 – 4/52	1	2/52 – 4/52
Move time (in years)	0.5/52 – 4/52	1	2/52 – 4/52
# Components per resource	2 – 6	3	2 – 3

Table 7: Values that vary over a range