A Minimum Cost Flow model for Level of Repair Analysis

R.J.I. Basten∗, M.C. van der Heijden, J.M.J. Schutten

University of Twente, School of Management and Governance,
Department of Operational Methods for Production and Logistics,
P.O. Box 217, 7500AE, Enschede, Netherlands

August 20, 2008

Abstract
Given a product design and a repair network for capital goods, a level of repair analysis determines for each component in the product (1) whether it should be discarded or repaired upon failure and (2) at which location in the repair network to do this. In this paper, we show how the problem can be modelled as a minimum cost flow problem with side constraints. Advantages are that (1) solving our model requires less computational effort than solving existing models and (2) we achieve a high model flexibility, i.e., many practical extensions can be added. Furthermore, we analyse the added value of modelling the exact structure of the repair network, instead of aggregating all data per echelon as is common in the literature. We show that in some cases, cost savings of over 7% can be achieved. We also show when it is sufficient to model the repair network by echelons only, which requires less input data.

Keywords: Maintenance, Supply chain management, Level of repair analysis, Mixed integer programming, Minimum cost flows

1 Problem setting and literature
For capital goods, such as military naval equipment, MRI-scanners, or trains, customers increasingly take total life cycle costs (LCC) into account in their purchasing decisions (Ferrin and Plank, 2002). Also, we see a trend that customers outsource activities for system upkeep to the original equipment manufacturer (OEM) using service contracts that guarantee a certain service level against fixed yearly costs. For the OEM, this can be attractive, since selling services is generally more profitable than selling products (Deloitte, 2006; Murthy et al., 2004; Oliva and Kallenberg, 2003). This means that it becomes important for the OEM to take the costs of maintenance into account when designing new products. Costs of more reliable components can be earned back by lower maintenance costs during the product life cycle.

Generally, capital goods are repaired by replacement, which means that a failed component is removed from the system and replaced by a functioning one. A defective component can either be discarded or repaired. If it is discarded, a new component needs to be purchased. If the component is repaired, this often means that a subcomponent failed and is replaced by a functioning one. The subcomponent should in turn be repaired or discarded itself. The system is thus seen as a multi-indenture system such as shown in Figure 1.

If a component should be repaired, it should also be decided where to do that. For example, if we consider military naval equipment, repairs can be performed on board the ship, at its marine base, a central depot, or even at the OEM. A network that connects all ships, bases, depots, and the OEM is called a multi-echelon repair network, see Figure 2 for an example.

∗Corresponding author. Tel.: +31 53 489 4443; fax: +31 53 489 2159
E-mail addresses: r.basten@utwente.nl, m.c.vanderheijden@utwente.nl j.m.j.schutten@utwente.nl.
In an early stage of the product life cycle, decisions upon discard or repair, and location of maintenance activities should be taken. These decisions are covered by a level of repair analysis (LORA). For a given product design and a repair network, a LORA determines for each component (1) whether it should be discarded or repaired upon failure and (2) where to do that, with the goal to minimize the life cycle costs. Relevant costs can be classified as fixed or variable in the number of failures that occur. Fixed costs include costs for resources such as test equipment and tools. Variable costs include costs for working hours of service engineers and transportation of components. Spare parts can be modelled by (a combination of) fixed and variable costs.

The LORA problem is nontrivial, due to the:

- The multi-indenture system structure: Decisions for a subcomponent are only needed if the parent component is repaired.
- The multi-echelon repair network: If a component is repaired at a certain location, it is illogical to repair a subcomponent at a location downstream in the network.
- Shared resources: Some components share resources that are needed to repair them. This means that it is often cost effective to repair these components at the same location.

Surprisingly, the scientific literature on LORA is limited. Barros (1998), Barros and Riley (2001), Saranga and Dinesh Kumar (2006) and Basten et al. (2008a) all assume infinite capacity of resources and aggregate all data per echelon. This means that every three echelon repair network would be represented as in Figure 3. The key differences between these papers are:

- Barros (1998) and Barros and Riley (2001) assume that all components at one indenture level share the same resource.
- Saranga and Dinesh Kumar (2006) assume that components do not share resources and every component needs exactly one resource.
- Basten et al. (2008a) relax the assumptions made in the forementioned papers in that sets of components sharing resources can be defined freely.

Brick and Uchoa (2007) do not aggregate all data per echelon level, but they model only 1 echelon and effectively assume 2 indenture levels. Integrated in their LORA is the decision of which facilities to open.

In this paper, we model the LORA problem as a minimum cost flow model with side constraints. As we will see, we can solve problem instances much faster using our new formulation than using the formulation of Basten et al. (2008a). We compare our model with theirs, because it is the most generic model that can cope with multi-indenture systems and multi-echelon repair networks. Furthermore, our formulation allows for many extensions in an elegant manner, including repair probabilities, no-fault-found probabilities, and equipment with finite capacity. Although these extensions are the focus of an accompanying paper (Basten et al., 2008b), we analyse in this paper one generalization, namely modelling the exact structure of the repair
network instead of aggregating all information per echelon. This means that we can differentiate between the repair networks in Figures 2 and 4, which may lead to different decisions and costs. For example, for a European OEM, we may decide to repair certain components at the OEM for the installed base in Europe, whereas we repair the same components for the installed base in Asia at a local Asian repair shop. In this way, we avoid transportation costs and spare parts costs due to shorter lead times at the expense of required resources at the local repair shop.

A drawback is that we need more information of the structure of the repair network and the assignment of the installed base to the various locations. This information can be hard to obtain early in the product life cycle when a LORA is performed. In this paper, we analyse the impact of modelling the exact repair network in terms of reduction of life cycle costs. We find that in repair networks that are unbalanced in the locations, such as shown in Figure 4, significant cost reductions of sometimes more than 7% can be achieved. We also find that in repair networks that are balanced in the locations, no significant cost reductions can be achieved by modelling the exact network, even if costs differ across the repair network. In those cases, it is not worth the additional effort to collect input data.

Summarizing, the added value of this paper to the literature is two-fold:

• We provide a new minimum cost flow model for the LORA problem that is fast and flexible.
• We provide insight in the impact of modelling the exact repair network on the life cycle costs and show under which conditions data aggregation per echelon is sufficient.

The structure of the remainder of this paper is as follows. In Section 2 we explain which inputs a LORA needs and what assumptions we make. In Section 3 we model the LORA problem as a minimum cost flow model with side constraints. Section 4 gives the results of our experiments. We end with conclusions and directions for further research in Section 5.

2 Model assumptions and input data

We model a multi-indenture system structure, with \( F \) being the set of all components. The subcomponents or children of a (father) component \( f \in F \) form the set \( \Gamma_f \). For example, in Figure 1: \( \Gamma_1 = \{2, 3\} \). For each component, there are three possible decisions \( d \in D \) at every location: Discard, repair and move. Discard means that the component is scrapped and a new one should be bought. Under the repair option, we assume that any repair is successful and that a component on which a repair is performed, has really failed (in practice, a ‘no-fault-found’ may occur), although both assumptions can be generalized (see Basten et al., 2008b). The move option means that the component is moved to the location at the next higher echelon for further decision making. Note that the ‘move’ option does not exist at the highest echelon level, the central depot.

A system in the installed base is always repaired by replacing a subsystem. Therefore, the first indenture level that we model is that of the subsystems (also called line replaceable units or LRU's): Component 1 in Figure 1. These components form the set \( F_s \). A failure in component \( f \) is due to a failure in subcomponent \( g \) with probability \( q_{fg} \). In that case, repairing component \( f \) means replacing subcomponent \( g \). For subcomponent \( g \), we have to decide whether to discard, repair or move it. It is possible that two (or more) subcomponents fail at the same time, which could mean that for some component \( f \): \( \sum_{g \in \Gamma_f} q_{fg} > 1 \). There is also a probability \( q_{ff} \) that a failure can be repaired directly, without replacing a subcomponent. No further decisions have to be taken if a component is repaired directly.

We model a multi-echelon repair network, with \( L \) being the set of all locations. The structure is divergent, that is, each location in the network has a single upstream location to which it can move its failed components. All locations that supply failed components to location \( l \) form the set \( \Phi_l \). For example, in Figure 2: \( \Phi_1 = \{2, 3\} \). All the system locations (echelon 1), so the locations \( l \) that have \( \Phi_l = \emptyset \), form the set \( L_s \).

If a system fails, this is due to a failure in a subsystem, so components \( f \in F_s \). We define \( m_{fl} \) as the average yearly number of failures of component \( f \in F_s \) at location \( l \in L^s \). For each
failure, decisions have to be taken as described above. If a certain decision \( d \in D \) is chosen for component \( f \in F \) at location \( l \in L \), then variable costs of \( c_{v \text{fl}} \) are made each time this component fails.

We model resources \( r \in R \) with \( \Theta_r = \{(f,d) \mid \text{resource } r \text{ is required for decision } d \text{ for failures in component } f\} \). For example, \( \Theta_{r_1} = \{(f_1, \text{repair}), (f_1, \text{discard}), (f_2, \text{repair})\} \). We assume that the resource capacity is infinite, although we can extend our model to finite capacities (see Basten et al., 2008b). Without loss of generality, we have chosen to minimize the average total costs per year with our definition of \( m_{fl} \). Therefore, we define fixed costs \( c_{f rl} \) per year if we decide to use resource \( r \) at location \( l \). These costs are, for example, the yearly depreciation costs of the resource and costs of capital.

3 Minimum cost flow model

This section explains how the LORA problem can be modelled as a minimum cost flow model with side constraints (see, for example, Ahuja et al., 1993, for an overview of minimum cost flow models). To define the graph underlying the flow model, we need four different node types: Source nodes \( (v \in V^s) \) are used to represent the occurrence of failures in subsystems (indenture 1) at system locations (echelon 1). The flows from these source nodes arrive at decision nodes \( (v \in V^d) \) where a decision is made between the three available options: discard, repair and move. The variable costs are attached to the outgoing arcs of the decision node, each representing an option. If repair is chosen, then a transformation node \( (v \in V^g) \) is used to represent that a failure in a parent is due to a failure in any of the children. If no decisions need to be made anymore, the flow goes to a sink node \( (v \in V^t) \). We model the use of resources by side constraints on the minimum cost flow model: if the outgoing arc of a decision node represents an option for a component that can only be chosen if a resource is available, then the capacity of this arc is 0 if the resource is not available.

Section 3.1 explains how we construct the graph that forms the basis of the flow model. Section 3.2 shows how the resources are added to the model as side constraints. Section 3.3 provides the formal model formulation.

3.1 Construction of the graph

In this section, we explain how the node types are used in the model and we define the incoming and outgoing arcs \((v,w)\) and the relevant cost parameters. To illustrate our model, we use the following example throughout this section: We have a two-indenture system with three components \( (\Gamma_1 = \{2,3\}) \), see Figure 5) and a two-echelon repair network with three locations (see Figure 6). We show an example of each node type, and after we have introduced all the node types, we show the complete flow model. In all figures related to this example, a number next to an arc represents a component, whereas a letter next to an arc represents an option (r=repair, d=discard, m=move).

3.1.1 Source nodes

Source nodes represent occurrences of failures of a certain subsystem at a certain system location. In our example only failures in component 1 occur at locations 2 and 3. This means that we have two source nodes in our flow model, of which the one for location 2 is shown in Figure 7.
Figure 7: Source node

![Source node diagram]

*Failures of component 1, shown by the number next to the arc, originate at the source node.

Figure 8: Decision node

![Decision node diagram]

*Failures of component 1 at location 2 go into the decision node, represented as ‘1,2’ in the node. Options repair ‘r’, discard ‘d’, and move ‘m’ can be chosen.

The flow out of this source node is equal to the yearly number of failures of component 1 at location 2: \( m_{1,2} \). In general, for every component \( f \in F_s \) and every location \( l \in L_s \), there is one source node \( v \in V_s \) with outflow \( s_v := m_{fl} \).

### 3.1.2 Decision nodes

If a subsystem at a system location fails, there are three options to choose from:

- Move the component to the next higher echelon.
- Repair the component, which means replacing a subcomponent or repairing the component directly.
- Discard the component.

In the flow model, it means that an arc originating at a source node terminates at a decision node \( v \in V^d \). Every arc going out of the decision node represents one of the available options.

Figure 8 shows the decision node for component 1 at location 2. If arc \((v, w)\) represents the repair option ‘r’, then the variable costs for using this arc, \( c_{vw}^a \), are equal to the variable costs of option ‘repair’ for component 1 at location 2: \( c_{1,2}^{\text{repair}} \). In general, the variable costs, \( c_{vw}^a \), for using an arc \((v, w) \in V^d\) are equal to \( c_{fl}^X \), where arc \((v, w)\) represents decision \( d \) for component \( f \) at location \( l \). In this way, the variable costs of the LORA model are attached to the arcs originating at the decision nodes; all other arcs have zero costs associated to them.

If the decision is taken to move component 1 from location 2 to location 1, a decision has to be taken at location 1. This means that the arc representing the move option ‘m’ in our example (Figure 8) terminates at the decision node representing component 1 at location 1. In general, the arc representing the move decision for component \( f \) at location \( k \) terminates at the decision node representing component \( f \) at location \( l \), where \( k \in \Phi_l \) (\( \Phi_1 = \{2, 3\} \)). Note that at a node representing a component at the highest echelon, location 1 in our example, the ‘move’ option is not available. The arcs representing the repair and discard options are discussed below.

### 3.1.3 Transformation nodes

Transformation nodes represent the repair of a parent component. A flow representing failures in a parent component goes into the node and is split according to the probability that a failure in that parent component is due to each specific child component. Each outgoing arc terminates at a decision node for a child component. In our example, a failure in component 1 can be caused by a failure in component 2 or 3 (\( \Gamma_1 = \{2, 3\} \)). The arc representing the repair decision ‘r’ for component 1 at location 2 (in Figure 8) terminates at the transformation node that is shown in Figure 9. The two arcs originating at the transformation node represent failures of components 3 and 2 respectively. Suppose that 50% of those failures are caused by failures in component 1 and 50% are caused by failures in component 2, then the variable costs attached to the arcs are equal to 0.5 \( c_{fl}^X \) for each arc, respectively.

\( ^1 \)The source node could also be integrated in the decision node, but for clarity we prefer to have a distinction between source and decision nodes.
The flow going into the node results from a repair 'r' of component 1 at location 2, shown as '1,2,r' in the node. Components 3 and 2 flow out of the node respectively.

The flow that is discarded 'd' goes into the sink.

Component 2 and 40% are caused by failures in component 3, which means that 10% of the failures is repaired directly ($q_{1,2} = 0.5, q_{1,3} = 0.4, q_{1,1} = 0.1$). If the outgoing arcs $(v, w)$ and $(v, u)$ represent components 3 and 2 respectively, then $g_{vw} = 0.4$ and $g_{vu} = 0.5$. Since no further decisions need to be taken for the failures that are repaired directly, it is not necessary to model them.

In general, transformation nodes $v \in V^g$ represent the repair of a parent component $f$ by replacement of any of the subcomponents $g \in \Gamma_f$ at location $l$. If arcs $(v, w)$ represent components 3 and 2 respectively, then the factor of inflow in node $v$ that flows out on arc $(v, w)$, $g_{vw}$, is equal to $q_{fg}$.

### 3.1.4 Sink nodes

If no other decisions need to be taken after a certain decision has been made, then the flow goes to a sink node. In the example, arcs that terminate at a sink node represent the decision ‘discard’ for any component at any location and the decision ‘repair’ for component 2 or 3 at any location. Figure 10 shows a sink node.

### 3.1.5 Example

We already showed parts of the flow model that results from the LORA problem that we used as an example throughout this section. Figure 11 shows the complete resulting flow model. The dotted arcs for component 3 should be replaced by flows similar to the flows for component 2. We left them out to improve the readability of the figure.

### 3.2 Modelling resources as side constraints

In some cases, resources $r$ are needed before a certain decision can be taken for certain components. For example: Resource $r = 2$ may be needed if and only if component 1 is to be repaired. In that case, $\Theta_2 = \{(1, \text{repair})\}$. At all three locations in our example repair network, we can decide whether we want to buy resource 2. Therefore, we distinguish in the flow model three so called ‘flow resources’ $e = 1, 2, 3$ that represent resource 2 at location 1, 2, 3 respectively. If flow resource 3 is available, then the arc representing repair of component 1 at location 3 is enabled. This means that $\Omega_3 = \{(v, w) \mid \text{arc } (v, w) \text{ denotes decision } \text{‘repair’ for component 1 at location 3} \}$. The costs of enabling flow resource 3 ($c_{3l}$) are the yearly costs for resource 2 at location 3: $c_{3l}$. In general, for each location $l$ and every resource $r$ there will be a flow resource $e$ in the model such that $\Omega_e = \{(v, w) \mid \text{arc } (v, w) \text{ denotes decision } d \text{ for component } f \text{ at location } l \text{ with } (f, d) \in \Theta_r \}$ and $c_e := c_{el}$. Note that fixed costs are only related to the arcs originating at the decision nodes.
3.3 Flow model formulation

In our flow model, we have two types of decision variables:
- \( X_{vw} \), the amount of flow through arc \((v, w)\).
- \( Y_e \), a binary variable that indicates whether flow resource \( e \) is bought.

Let \( A \) be the set of all arcs and \( E \) the set of all flow resources. The resulting model formulation is:

\[
\text{minimize } \sum_{(v, w) \in A} c_{vw}^a \cdot X_{vw} + \sum_{e \in E} c_e \cdot Y_e
\]  

subject to:

\[
X_{vw} = s_v, \forall (v, w) \in A \mid v \in V^s
\]  

\[
\sum_{u \mid (u, v) \in A} X_{uv} = \sum_{w \mid (v, w) \in A} X_{vw}, \forall v \in V^d
\]  

\[
X_{vw} = g_{vw} \cdot \sum_{u \mid (u, v) \in A} X_{uv}, \forall (v, w) \in A \mid v \in V^g
\]  

Constraint 2 states that the outflow of each source node \( v \) is equal to \( s_v \) and Constraint 3 assures that the inflow into any decision node is equal to the outflow. For any arc \((v, w)\) going out of a
transformation node, Constraint 4 assures that the total inflow into that transformation node is transformed to outflow on arc \((v, w)\). Constraint 5 assures that only arcs are used that are enabled due to the availability of flow resources; they become uncapacitated. Arcs that are not in any \(\Omega_e\) are not capacitated either. For each arc \((v, w)\) that is part of any \(\Omega_e\), we set the value of the big M equal to the maximum possible value of \(X_{vw}\).

4 Computational experiments

In Section 4.1 we compare the time it takes to solve problem instances using the flow model and the model of Basten et al. (2008a). A more efficient optimization method is useful to solve large problem instances, but it is also useful if a LORA is performed multiple times. This is for example the case, if an iterative procedure is used to solve a LORA and spare parts optimization simultaneously. Section 4.2 compares modelling the repair network exactly and aggregating all inputs per echelon. We compare both the time it takes to solve problem instances and the cost reductions that can be achieved by modelling the repair network exactly.

For our tests, we generate instances of the LORA problem and solve these using the CPLEX callable library version 11 (with default settings), running under windows XP, service pack 2, on an Intel Centrino Duo, 2 GHz with 2 GB ram. Although CPLEX 11 can use both cores of the dual core processor, it seldomly does for these problems.

4.1 Comparison with the model of Basten et al. (2008a) in terms of optimization time

We generate problem instances in which all data is aggregated per echelon, corresponding to the assumptions of Basten et al. (2008a). In our model, this is equivalent to a network structure with one location at each echelon. We use the problem instance generator as described by Basten et al. (2008a). We omit the details here for sake of simplicity.

We vary the four input parameters that most heavily influenced the optimization time in their paper: The number of components, echelons and indenture levels, and the maximum number of resources per component. If the maximum number of resources per component is 2, this means that in order to repair a component, at most 2 different resources are required. See Table 1 for the settings. For each combination of parameters, we generated 25 problem instances. In total, this makes \(4 \cdot 2 \cdot 2 \cdot 2 \cdot 25 = 800\) test runs.

<table>
<thead>
<tr>
<th>Parameter (varied)</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td># Components</td>
<td>500 – 1,000 – 2,000 – 5,000</td>
</tr>
<tr>
<td># Echelons</td>
<td>2 – 3</td>
</tr>
<tr>
<td># Indenture levels</td>
<td>2 – 3</td>
</tr>
<tr>
<td>Max. # resources per component</td>
<td>1 – 2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter (not varied)</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td># Resources</td>
<td>100</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter (not varied)</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yearly demand</td>
<td>0.05 – 5</td>
</tr>
<tr>
<td>Variable costs</td>
<td>50 – 1,000</td>
</tr>
<tr>
<td>Fixed costs</td>
<td>500 – 10,000</td>
</tr>
</tbody>
</table>

Table 2 shows the average time it takes to solve the LORA problem instances for each parameter that we varied. It is clear that our model increasingly outperforms the model of Basten et al. if the number of components, indentures or echelons or the maximum number of resources per component increases.
4.2 Effect of modelling the exact repair network

In this section, we show the circumstances under which exactly modelling the repair network reduces the total costs and those under which inputs can be aggregated. We also compare the time it takes to solve these problems.

We generate instances that are realistic in practice, based on our experience at Thales Nederland, a manufacturer of naval sensors and naval command and control systems. We make comparisons (1) for a base situation with a symmetrical repair network, and for an asymmetrical network (2) in terms of the number of system locations that is attached to each intermediate depot and (3) in terms of the costs for moving and repairing components. Section 4.2.1 explains the problem instances we use and Section 4.2.2 discusses the results.

4.2.1 Problem instances

In all tests, the system structure consists of 25 components at the first indenture (subsystems), 125 at the second level and 625 at the third level. We use random generators to construct 10 instances of a product structure with corresponding failure rates and cost factors, see Appendix A for details.

For our tests, we use various repair networks, of which some are balanced in terms of the number of system locations per intermediate depot and some are not. We call them balanced and unbalanced in the locations. The smallest balanced network consists of one central depot, two intermediate depots and four system locations (two per intermediate depot), see Figure 2. We vary the number of intermediate depots (2 or 10) and the number of system locations per intermediate depot (2 or 10).

In the unbalanced networks, there are 2 system locations per intermediate depot for half of the intermediate depots and 10 system locations per intermediate depot for the other half of the intermediate depots. Below, we call the left half of the intermediate depots with the attached system locations the ‘left half’. The other intermediate depot(s) with the attached system locations are called the ‘right half’. This holds for both balanced and unbalanced networks.

Besides being unbalanced in the locations, repair networks can be unbalanced in terms of the costs. In our tests, costs in the left half and at the central depot are always equal. Repair and move costs in the right half can differ from the costs in the left half and at the central depot. We test what happens if the repair costs in the right half are 0.5 or 2 times the repair costs in the left half (and at the central depot). We say that the relative repair costs are 0.5 or 2. In
the same way, we test with relative move costs of 0.25 or 4. These values are chosen because for a European OEM, the costs of moving components to Asia can be a number of times as high as those costs in Europe. Repair costs can differ as well, but the relative difference is assumed to be smaller. Discard costs are assumed to be approximately the same at all locations, since a main part of these costs are due to the costs of buying a new component.

There are 10 or 25 resources (types of test equipment) and we distinguish two cases for the number of resources that each component requires. In the first case, 70% of the components needs no resources, 20% needs one resource, and 10% needs 2 resources: '0.7–0.2–0.1'. The other case is '0.25-0.5-0.25'.

Summarizing, we have 7 experimental factors, namely the number of intermediate depots, the number of system locations in the left half, the number of system locations in the right half, the relative move costs in the right half, the relative repair costs in the right half, the total number of resources, and the number of resources per component. This gives \(2 \cdot 2 \cdot 2 \cdot 3 \cdot 2 \cdot 2 = 288\) combinations. For each combination, we generate 10 problem instances as described in Appendix A. In this way we prevent that we draw conclusions based on one exceptional case.

### 4.2.2 Results

Table 3 shows for the 8 repair networks, the average time it takes to solve the exact and aggregated model. It also shows the average and maximal difference in optimal solution value for both models. In each test we vary over all other parameters, as explained in the previous section. We focus on the other parameters below.

| #Intermediate
depots | # System locations per internm. depot | Optimization time | Cost difference | # Aggr. |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Left half</td>
<td>Right half</td>
<td>Exact</td>
<td>Aggr.</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>10</td>
<td>2.0s.</td>
<td>0.1s.</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>2</td>
<td>1.7s.</td>
<td>0.1s.</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>10</td>
<td>1.6s.</td>
<td>0.1s.</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>2</td>
<td>1.0s.</td>
<td>0.1s.</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>10</td>
<td>8.1s.</td>
<td>0.1s.</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>2</td>
<td>8.4s.</td>
<td>0.1s.</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>10</td>
<td>10.2s.</td>
<td>0.1s.</td>
</tr>
</tbody>
</table>

"Aggr." means the aggregated network. It may seem strange that there is a difference between row 2 and 3 (2 intermediate depots and unbalanced network). However, these cases are not the same because, for example, if the relative repair or move costs are not 1, this affects more system locations in row 2 than in row 3.

We see that the aggregated models can be solved much faster than the exact models. Still, it took only 104 seconds to solve the most time consuming problem instance using the exact model. This means that the exact model can be solved fast enough to be used in practice.

The maximum cost difference for networks that are balanced in the locations is 1.2% and only 8% of these problem instances lead to a cost difference of more than 0.5%. Because modelling the exact network requires more inputs, there is hardly any reason to do this if the repair network is balanced in the locations, even if the network is unbalanced in the costs. However, in our tests, the spare parts costs that are added to the variable costs, are always balanced in the network, even if the remainder of the variable costs for repair and move are unbalanced. In a global repair network, the spare parts costs may be unbalanced too, due to differences in lead times. If we vary the spare parts costs with the move and repair costs in a network that is balanced in the locations, we see cost differences between the exact and aggregated model of over 5%. However, our basic way of incorporating spare parts, does not allow us to analyse this in detail.
The maximum cost difference for networks that are unbalanced in the locations is 7.4%. For 29% of the problem instances, a cost difference of more than 2.5% is achieved and 6% of the problem instances leads to a cost difference of more than 5%. We focus in more detail on the networks that are unbalanced in the locations in Table 4, in which we vary the three parameters that most heavily influence the cost differences: The number of intermediate depots, the number of resources, and the number of resources per component. We see that high cost differences are mainly achieved in problem instances with 10 intermediate depots. If there are also many resources in total and many components need 1 or 2 resources, a large cost reduction by modelling the exact network is almost guaranteed. However, we cannot define a broad category of problem instances in which we can guarantee that there are no cost reductions possible.

<table>
<thead>
<tr>
<th>#Intermediate depots</th>
<th># Resources</th>
<th>Cost difference exact – aggregated</th>
<th>Mean</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total</td>
<td>Per comp.</td>
<td>Mean</td>
<td>Max</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>0.7–0.2–0.1</td>
<td>1.1%</td>
<td>3.9%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.25–0.5–0.25</td>
<td>0.1%</td>
<td>1.2%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.7–0.2–0.1</td>
<td>0.5%</td>
<td>1.9%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.25–0.5–0.25</td>
<td>2.4%</td>
<td>5.1%</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>0.25–0.5–0.25</td>
<td>1.1%</td>
<td>3.9%</td>
</tr>
<tr>
<td></td>
<td>0.7–0.2–0.1</td>
<td>2.4%</td>
<td>6.1%</td>
<td>6.9%</td>
</tr>
<tr>
<td></td>
<td>0.25–0.5–0.25</td>
<td>4.0%</td>
<td>7.4%</td>
<td>7.4%</td>
</tr>
</tbody>
</table>

Table 4: Comparison of three most important parameters

*a*‘Per comp.’ means per component. ‘Aggr.’ means the aggregated network.

If relatively small cost reductions can be achieved by modelling the exact network, the solutions of the exact model and the aggregated model can still differ substantially. To give an example, we focus on the problem instances in which the repair network is unbalanced in the locations, there are 10 intermediate depots, the number of resources per component is 0.25–0.5–0.25, and there are 10 resources. Table 5 shows that for these problem instances, the costs of resources are 19% higher on average if the model is aggregated. However, this is compensated for by lower variable costs, so that the total costs differ only 2.1%.

<table>
<thead>
<tr>
<th>Type of model</th>
<th>Variable costs</th>
<th>Costs of resources</th>
<th>Total costs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exact</td>
<td>43,900,000</td>
<td>12,400,000</td>
<td>56,300,000</td>
</tr>
<tr>
<td>Aggregated</td>
<td>42,200,000</td>
<td>15,300,000</td>
<td>57,500,000</td>
</tr>
<tr>
<td>Difference</td>
<td>-4.0%</td>
<td>19.0%</td>
<td>2.1%</td>
</tr>
</tbody>
</table>

Table 5: Example of different solutions

If there are multiple, really different solutions that lead to approximately the same total costs, there can be other, more qualitative reasons to choose for another solution than the one with the lowest total costs. Industry might be interested in tools that can provide these different solutions.

We conclude that modelling the repair network exactly brings cost reductions of almost 2% on average for networks that are unbalanced in the locations. In some cases, the cost reductions are over 7%, which means that it is worthwhile to model the repair network exactly for unbalanced networks. For networks that are balanced in the locations, cost reductions are never higher than 1.2%, which means that it is doubtful if the additional effort of acquiring all inputs is worth it. In this case, aggregating all data does not lead to much higher costs. However, we do note that if the costs for spare parts are not equal in the whole network, due to a difference in lead time between different parts of the global repair network, it might be necessary to explicitly model the repair network. However, more research is needed on the integration of spare parts optimization into the LORA before this question can be answered.
5 Conclusions and directions for further research

We have modelled the LORA problem as a minimum cost flow model with side constraints. This formulation allows us to model all kinds of extensions in an elegant manner. Such extensions include repair probabilities, no-fault-found probabilities and equipment with a finite capacity. Besides that, we have shown that the LORA problem with all data aggregated per echelon level, can be solved much faster using our formulation than it could using existing formulations (a factor 5 on average for large problem instances).

Our model allows us to explicitly model the repair network instead of aggregating all information per echelon level, as is often done in the literature. We have shown that with networks that are unbalanced in the locations, cost reductions of over 7% can be achieved by modelling the exact network. If networks are balanced in the locations, then the maximal costs reductions that can be achieved are only 1.2%, even if the network is unbalanced in the costs. However, our research suggests that if lead times differ across the repair network, significant cost reductions may be achieved by modelling the exact network, even for networks that are balanced in the locations. To be able to analyse this, integration of spare parts optimization in the LORA is necessary.

We have shown that in some cases, only small cost differences exist between using the exact and the aggregated network, although the decisions that are taken differ a lot. If there are multiple ways to achieve almost the same total costs, there can be other, more qualitative reasons to choose for another solution than the one with the lowest total costs. Future research could lead to an approach that results in multiple alternative solutions that differ a lot in terms of decisions, but lead to almost the same optimal solution value.

Acknowledgments

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.

References


Appendix A

In this appendix, we describe the random generator that we used in Section 4.2.1 to generate product structures, failure rates and cost factors.

In all tests, the system structure consists of 25 components at the first indenture (subsystems), 125 at the second level and 625 at the third level. Every second and third indenture component is randomly attached to a lower indenture component, so a first indenture component can have zero subcomponents. The demand per subsystem (first indenture) is drawn from a uniform distribution on the interval [0.01, 1]. The percentage of demand for a parent that is due to a failure in a specific child is drawn from a uniform distribution on the interval [0.5/(number of children), 1.25/(number of children)], with a maximum of 1.

For each component, we draw a net price, excluding the costs of the children, from a shifted exponential distribution with shift factor 1,000 and rate parameter \( \lambda = \frac{7}{(100,000 - 1,000)} \). In this way, on average 1% of the components get a value larger than 100,000. We draw a new price for these components to avoid odd problem instances. The reason for our choice is that most systems have a large diversity of items in price, but there are considerably more cheap items than expensive items. The cheapest items (in our case items with a price below 1,000) are usually omitted from a regular LORA, because they are discarded by default.

Using these prices, we calculate the variable costs as follows:

- Repair costs as a fraction of the net price are drawn from a uniform distribution on the interval [0.1, 0.4].
- We recursively add the price of each child to its parent to get the gross item price. We do this after calculating the repair price, since repair of a parent means replacement of the child that was defective and taking a decision for the child, thus incurring costs for the child. Discarding or moving a parent does not lead to a decision, and thus costs, for the children.
- The discard costs as a fraction of the gross item price, including children, are drawn from a uniform distribution on the interval [0.75, 1.25]. 100% would be just the costs of replacing a defective component by a new one. However, on the one hand there may be disposal costs, on the other hand, some parts of a defective component can be recycled or re-used.
- The move costs are always 1% of the gross item price.

We also need to take spare part costs into account. If we neglect the spare parts costs, we see that resources are only bought at the central depot. The explanation is that if a resource is used at the central depot, only one resource needs to be bought. However, multiple resources are needed if it is used at a lower echelon level, since one resource needs to be bought at every location at that echelon. In practice, repairing everything at the central depot means that the availability of the systems goes down, which should be compensated for by buying spare parts.

Integrating spare parts optimization into the LORA is interesting future research, but it does not fit in the scope of this paper. However, there are multiple approaches for adding spare parts costs in a basic way. The approach that we have chosen is based on the difference in lead times for the different options that can be chosen for each component. We assume that repair at the system location or at the intermediate depot takes a month, whereas repair at the central depot takes three months. Discard (and buying a new item) has a lead time of half a year. Moving
a component to a higher echelon leads to an additional lead time of half a month. Using these lead times, we estimate spare parts costs by multiplying the demand for the component, the leadtime, a safety factor of 2, the price of the component and holding costs of 30%. A correct item approach would use the standard deviation of the demand instead, but this would make our model non-linear. A correct system approach, such as the metric-like approaches (see, for example, Sherbrooke, 2004), would be even more problematic. We tested other approaches and other safety factors, but this approach leads to reasonable results.

The price of each resource is drawn from a shifted exponential distribution with shift factor 10,000 and rate parameter $\lambda = 7/(1,000,000 - 10,000)$ in the same way as described above for the prices of the components. As a result, prices vary between 10,000 and 1,000,000. We randomly assign components to resources, such that the percentage of components that needs 0, 1 and 2 resources are 70%, 20% and 10% respectively in the first case, and 25%, 50% and 25% in the second case.