Chapter 2

Transshipments and Repositioning in the Standard Single-echelon Rental System

Abstract. This chapter considers the standard single-echelon rental system applied to the setting of libraries. Two types of exchanges take place: transshipments from library to library to fulfill requests for items and repositioning operations to redistribute items between libraries. We determine optimal decisions for transshipments and repositioning, so that logistic costs are minimized. In current practice, libraries typically send transshipped items back to their original library after the rental period, i.e., there is fixed ownership. We consider a more general policy, in which it is possible to reposition items to any other library. By means of stochastic dynamic programming, we derive the optimal policy for small instances. For larger instances we present two heuristics: the cluster and the expected shortage reduction (ESR) heuristic. By comparing optimal policies and heuristics with policies currently applied in practice, we quantify the cost of having fixed ownership. Experiments indicate that fixed ownership can increase logistic costs by over 30%, hence we propose cost-efficient and easy-to-coordinate alternatives for stock repositioning.

2.1 Introduction

As discussed in Chapter 1.2, libraries are increasingly adopting online ordering systems. Instead of going to a physical library, customers can order books online and have these delivered to a library, in a pick-up locker, or at home, on a given day of the week. This reduces expenses on personnel and buildings since libraries do not have to be open full-time, or can be closed entirely. Operational costs, on the other hand, may increase since more items have to be transshipped. These transshipments are costly because the library employees have to spend time on retrieving and storing items. Additionally, costs are incurred for vehicles, fuel, and hiring of transportation personnel. If the libraries do not coordinate the transshipments appropriately, logistics may become a burden on their budget.

In this chapter we therefore present an optimal policy for coordinating the transshipments and repositioning of library items. In the case of item requests, we determine optimal libraries from which to transship. Moreover, we define a more general policy for repositioning items after the rental period than used in practice. Instead of repositioning transshipped items to fixed owner locations after the rental period, we consider the option of keeping items at the receiving library, or even sending items to libraries that are likely to receive demand in the future. We show that this repositioning policy can lead to significant costs savings. For a general number of libraries, a dynamic programming approach is presented to determine cost-minimizing transshipments and repositioning. For every state of the system and every observed demand, the transshipment and repositioning decision that leads to the lowest expected future costs is determined. As is to be expected, the computation times of the optimal procedure grow exponentially in the input. Hence, heuristic procedures are proposed for models of larger size. Our proposed expected shortage reduction (ESR) heuristic performs near-optimal and yields significant improvements compared to current practice. While the main focus of this chapter is on library items, the approach may also extend to other multi-location rental systems with transshipments such as DVD and tool rentals.

The coordination of transshipments in libraries is a subject that, to the best of our knowledge, has not been dealt with in scientific literature so far. Research on libraries has mainly focused on two directions. The first direction is predominantly financial: it concerns the purchase of books and budget allocations for libraries (see,
Transshipments and Repositioning in the Standard Single-echelon Rental System

17

Transshipments and Repositioning in the Standard Single-echelon Rental System 17

e.g., [Allen et al., 2003; Chan, 2008]. The second direction is social: the composition of
the user-base and lending behavior of customers is studied (see, e.g., [Boter & Wedel,
2005; Løyland & Ringstad, 2011]. The operational aspects of managing a library and
costs thereof seem to have a less prominent role in the literature. Transshipments,
which are also known as interlibrary loans, are gaining serious interest (Corthouts
et al., 2011). The coordination of these interlibrary loans is not covered, however.
Our study gives substance to the interest from practice by providing optimal rules for
coordination.

More generally, the problem of coordinating shipments between libraries can be
regarded as an inventory problem with lateral transshipments (Paterson et al., 2011).
Three important aspects from the literature characterize the problem: reactive and
preventive lateral transshipments, and reverse logistics (Fleischmann et al., 1997; Shi
et al., 2011). The reactive transshipments are shipments in immediate response to
demand, while the preventive transshipments concern redistribution of items after re-
turns. The reverse logistics part is that items return to the libraries after renting and
become available again for fulfilling demand of future customers. The combination of
these three aspects is, to our knowledge, unique. There are many articles that deal
with these aspects separately, such as finding the optimal coordination for reactive
transshipments (Herer et al., 2006) and proactive transshipments (Tiacco & Saetta,
2011). We have found only one article that discusses the combination of transshipping
and reverse flows (Ching et al., 2003). The library problem is probably most similar
to car rentals (George & Xia, 2011). However, as discussed in Chapter 1.3, reactive
transshipments are typically not carried out in car rental systems. A prominent solu-
tion technique in the above mentioned literature, and also the technique we apply, is
dynamic programming (Bellman, 1957). Dynamic programming has been successfully
applied before in inventory control (Agrawal et al., 2004; Olsson, 2009) and car rental
problems (George & Xia, 2011; Adelman, 2007).

The outline of the chapter is as follows. In §2.2 we explain the model and assump-
tions. In §2.3 we formalize the model and dynamic programming algorithm. In §2.4
and 2.5 we present the cluster and ESR heuristic. In §2.6 we show the results of the
dynamic programming algorithm and the comparison with the heuristics. In §2.7 we
conclude.
2.2 Problem Formulation

In this section we formulate the problem of transshipping and repositioning items in a multi-location library system. Time is discretized and consists of regular periods and repositioning periods. In regular periods libraries face demand, transship to meet demand, and receive returned items. These regular periods take place on a daily or weekly basis, depending on how the library system is organized. Repositioning is done at the start of every $T$ regular periods, for instance every week or month. The process of our problem is presented in the flowchart of Figure 2.1. At the start of the first period (now), the process starts with transition 1. A decision is made regarding how to reposition the items from the previous period over the libraries, in anticipation of future demands based on statistics of past demand. Because future demand is stochastic, this decision is nontrivial. The stock at the libraries after repositioning then forms the input for transition 2. In transition 2, the periodic random demand for each library arrives. This demand has to be satisfied from the available stock of items to the extent possible. If stock-outs occur while simultaneously items are available in another library, transshipments must be carried out in order to satisfy demand. Hence, this entails deciding which libraries to transship from and to. Once these transshipments are carried out, previously rented items return and become available for rental in the next period. Transition 2 is repeated $T$ times, after which the complete process repeats itself again starting from transition 1. The demands and returns during a period are the aggregates for the discretized time period. Demands are assumed to occur at the beginning of the transshipment period, since a client orders items online in advance to receive them in the new period. Returns occur at the end of the period, since returns remain unknown until items are actually returned.

Various assumptions are made in order to model the situation as described in Figure 2.1. These assumptions are listed below.

- The model considers the behavior of the system for a single item type. Repeated application of the model is possible for the full assortment.

- Each demand that can be fulfilled must be fulfilled. As the goal of the library is to fully serve its customers, every demand should be fulfilled when there are still items available in the system. Demand that cannot be fulfilled is lost.
Transition 1: repositioning

- Start with stock from the previous period
- Determine the repositioning decision for the available stock
- New stock after repositioning

Transition 2: transshipments

- Start with the stock after repositioning
- Demand arrives and is satisfied from available stock
- Determine the transshipments to fulfil excess demands
- Previously borrowed items return
- End with stock after transshipments and returns

Figure 2.1: Flowchart of the decision process

- Demand probabilities for the item are library dependent. Because each library has a different customer-base, demand probabilities are generally unequal.

- For each transshipped item a fixed unit transshipment cost incurred, independent of the distance between libraries. Fixed costs are realistic if transshipments are handled by mail \cite{Corthouts et al. 2011}, and also when the libraries use a fixed route for their transportation system. The truck will make a tour regardless of the transshipment decisions, and the only extra costs incurred are the handling costs per item.

- Each repositioning movement comes at a unit cost. The model allows for a difference in cost between transshipments and repositioning. Typically the costs for repositioning are lower, because repositioning is done during off-peak hours.

- Returns are geometrically distributed, i.e., an item has a fixed probability of returning in a given period. The geometric distribution captures part of the randomness in the return process while keeping the problem tractable through its memoryless property. In problems with DVDs or tool rentals, for which the product is often returned the next day, a geometric distribution with a high return probability is likely to be appropriate. The return probability is assumed not to depend on the library, since rental times are often very similar between libraries.

Most of the above assumptions are not very restrictive, except for the geometric returns. In the simulation study, however, we will show that the solution with geo-
metric returns leads to improvements over current practice even when returns follow an empirical distribution fitted from transaction data.

2.3 Stochastic Dynamic Programming Formulation

This section describes the dynamic programming algorithm for solving the transshipment and repositioning problem to optimality for a single item type. Figure 2.1 provides the guideline for this section. The notational style in this chapter draws partially from Powell & Topaloglu (2003), and the methodology for stochastic dynamic programming can be found in Puterman (2009). The positive and negative part functions, \((x)^+\) and \((x)^-\), used throughout the chapter, are defined as:

\[
(x)^+ = \max(x, 0), \quad (x)^- = -\min(x, 0),
\]

and are applied elementwise when \(x\) is a vector. First, we introduce the notation, and then define the state space and transitions between states.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(n)</td>
<td>Number of libraries.</td>
</tr>
<tr>
<td>(K)</td>
<td>Total number of items for the single item type.</td>
</tr>
<tr>
<td>(\bar{T})</td>
<td>Number of periods between consecutive repositioning transitions.</td>
</tr>
<tr>
<td>(T)</td>
<td>Time horizon, a multiple of (\bar{T}).</td>
</tr>
<tr>
<td>(t)</td>
<td>Index for the periods, (t = 1, \ldots, T).</td>
</tr>
<tr>
<td>(i)</td>
<td>Index for the libraries, (i = 1, \ldots, n).</td>
</tr>
<tr>
<td>(x_t)</td>
<td>Vector of available items at the libraries during period (t).</td>
</tr>
<tr>
<td>(y_t)</td>
<td>Vector of rented items at the libraries during period (t).</td>
</tr>
<tr>
<td>(S_t)</td>
<td>State at period (t), (S_t = (x_t, y_t)), for a discussion see §2.3.1.</td>
</tr>
<tr>
<td>(D_t)</td>
<td>Vector of random demand at the libraries at the start of the transshipment period (t).</td>
</tr>
<tr>
<td>(R_t)</td>
<td>Vector of random returns at the libraries in period (t).</td>
</tr>
<tr>
<td>(r)</td>
<td>Fixed probability for a return.</td>
</tr>
</tbody>
</table>
Transshipments and Repositioning in the Standard Single-echelon Rental System

2.3.1 State Space

To represent the state of the system, we require two quantities: the number of available and the number of rented items at each library for the single item type. Evidently, we keep track of available items because we model transshipments and repositioning of the available stock. To model the return process, knowledge of rented items is essential. Because clients return the item to their own library, and since items have a fixed probability of returning regardless of time rented, it suffices to know the number of rented items at each library. The state space, $S_t$, is thus characterized by the vector of available items and rented items at each library, $x_t$ and $y_t$. When the available stock at three libraries at time $t$ is given by $x_t = (1, 0, 1)$, and the number of rented items by $y_t = (0, 1, 1)$, the state space variable is $S_t = (x_t, y_t) = (1, 0, 1; 0, 1, 1)$. The sum of available and rented items always adds up to the total number of items $K$, i.e. $\sum_{i=1}^{n} x_{it} + \sum_{i=1}^{n} y_{it} = K$ for all $t = 1, \ldots, T$. In Figure 2.1 there are two transitions. In order to distinguish between states at these two transitions, the superscripts $E$ and $R$ are added to the notation. $S_t^E$ refers to the state after ‘emergency’ transshipments, while $S_t^R$ refers to the state after repositioning. We use the term emergency here, because these transshipments have to be done quickly in response to demand. In Olsson (2009) the state space is very similar: it consists of the items in stock and the items due.
2.3.2 Demand and Return Distribution

The vector $D_t$ of demands during period $t$ has a discrete distribution. For library $i, i = 1, \ldots, n$, its individual demand $D_{it}$ takes on the values $0, 1, \ldots, d$, where $d$ is the maximum demand that can be expected to occur at any of the individual libraries. The demand probabilities are given by $P(D_{it} = j) = p_{ij}$ for $i = 1, \ldots, n$ and $j = 0, \ldots, d$, with $\sum_{j=0}^{d} p_{ij} = 1$ for $i = 1, \ldots, n$. Demands are assumed to be independent between libraries and independent of the available stock. The probability of observing $D_t = (D_{1t}, \ldots, D_{nt})$ is the product of the individual library demand probabilities. In some cases, there may be correlated demand at the libraries, such as the demand for sports books in the period of the Olympic Games. The model presented here remains usable in such cases by using a dependent demand distribution that specifies the probability for each realization of $D_t$. For most regular items, however, this correlation is small, for which reason we will not treat it explicitly in the remainder of this chapter.

The vector of returns $R_t$ has a distribution depending on the number of rented items. Each rented item has a fixed probability of returning in period $t$, denoted by $r$. For each individual library, the total number of returns follows a Binomial($y_{it}, r$) distribution.

2.3.3 Transitions and Costs for Repositioning

Now we model the transitions and costs for repositioning, which relates to transition 1 from Figure 2.1. Suppose the period is $t$. In the repositioning period, the items from the previous period are repositioned. Let $S_{t-1}^{E} = (x_{t-1}^{E}, y_{t-1}^{E})$ be the state after the returns and transshipments from the previous period. Given the stock levels after returns, we decide how to move the stock between libraries. Let the decision variable for repositioning be denoted by $a^{R}_{it}$. It is a vector of mutations: when $a^{R}_{it}$ is positive, $a^{R}_{it}$ items are transshipped to library $i$. When $a^{R}_{it}$ is negative, $-a^{R}_{it}$ items are transshipped away from library $i$. After taking repositioning decision $a^{R}_{it}$, the vectors of available items, $x^{R}_{t}$, and rented items, $y^{R}_{t}$, are given by

$$x^{R}_{t} = x^{E}_{t-1} + a^{R}_{t}, \quad y^{R}_{t} = y^{E}_{t-1}$$

(2.1)
Transshipments and Repositioning in the Standard Single-echelon Rental System

with

\[ \sum_{i=1}^{n} a_{it}^{R} = 0, \text{ and } x_{t-1}^{E} + a_{t}^{R} \geq 0. \]  \hspace{1cm} (2.2)

The conditions in equation (2.2) assure that \( a_{t}^{R} \) is feasible. The sum of all mutations adds up to zero, since each transshipped item must arrive at another library. Moreover, the stock levels after repositioning should be nonnegative. Note that periods without repositioning can be covered through this transition as well, by setting \( a_{t}^{R} = (0, \ldots, 0) \).

The vector of rented items, \( y_{t-1}^{E} \), cannot be influenced, since the items are located at customers. The new state after repositioning is given by:

\[ S_{t}^{R} \equiv S_{t}^{R}(S_{t-1}^{E}, a_{t}^{R}) = (x_{t}^{R}, y_{t}^{R}). \]  \hspace{1cm} (2.3)

The cost for repositioning when choosing repositioning decision \( a_{t}^{R} \) is:

\[ C_{t}^{R}(S_{t-1}^{E}, a_{t}^{R}) = c_{t}^{R} \cdot \sum_{i=1}^{n} (a_{it}^{R})^{+}, \]  \hspace{1cm} (2.4)

i.e., the number of items repositioned is multiplied by the unit cost for repositioning. For example, suppose that the available stock is \( x_{t-1}^{E} = (3, 0, 1) \), and rented stock is \( y_{t-1}^{E} = (0, 2, 1) \). When making repositioning decision \( a_{t}^{R} = (-1, 1, 0) \), the new state is given by \( x_{t}^{R} = (2, 1, 1) \) and \( y_{t}^{R} = (0, 2, 1) \). In the example, one of the items in library 1 has been relocated to library 2, at cost \( c_{t}^{R} \).

### 2.3.4 Transitions and Costs for Transshipments

The transition for the transshipments is displayed as transition 2 in Figure 2.1. This transshipment transition includes demands, transshipments, and returns. Let the state after repositioning be \( S_{t}^{R} = (x_{t}^{R}, y_{t}^{R}) \). After demand, some libraries may have stock available, while others have excess demand. The transshipment decision shows how many items we should move away from and to each library to meet any possible demand. Let the decision variable for transshipments be denoted by \( a_{it}^{E} \). When \( a_{it}^{E} \) is positive, \( a_{it}^{E} \) items are transshipped to library \( i \), and when \( a_{it}^{E} \) is negative, \( -a_{it}^{E} \) items are transshipped away from library \( i \). The new vector of available stock, \( x_{t}^{E} \), after
demand $D_t$, transshipment decision $a_t^E$, and returns $R_t$, is given by:

$$x_t^E = x_t^R - D_t + a_t^E + m_t + R_t,$$

(2.5)

with

$$\sum_{i=1}^{n} a_{it}^E = 0,$$

(2.6)

$$a_t^E \geq -(x_t^R - D_t)^+,$$

(2.7)

$$a_t^E \leq (x_t^R - D_t)^-,$$

(2.8)

$$m_t = (x_t^R - D_t + a_t^E)^-,$$

(2.9)

$$\sum_{i=1}^{n} m_{it} = \left( \sum_{i=1}^{n} x_{it} - \sum_{i=1}^{n} D_{it} \right)^-.$$  

(2.10)

In equation (2.5), $x_t^R - D_t$ denotes the stock levels after demand and before transshipments. The decision variable $m_t$ is a nonnegative vector of unfulfilled demands, i.e., the lost demands. If $m_{it} > 0$ it means there are $m_{it}$ demands not fulfilled at library $i$. This variable is included for the situation when total demand exceeds available stock; if we do not include it, items disappear from the system. Finally, $R_t$ denotes the items returning at the end of period $t$.

The transshipment decision $a_t^E$ has to satisfy several restrictions, given in equations (2.6-2.10). Restriction (2.6) ensures that for each outgoing transshipment there is an incoming transshipment. Restriction (2.7) ensures that we transship at most the available stock, and zero in case of a stock-out. Restriction (2.8) models that only libraries with a stock-out receive a transshipped item. Restrictions (2.9) and (2.10) ensure that the total number of unfulfilled demands equals the amount by which total demand exceeds total stock. Together, the restrictions ensure that any possible demand is fulfilled.

The new vector of rented items, $y_t^E$, is given by:

$$y_t^E = y_t^R - R_t + D_t - m_t.$$  

(2.11)

The newly rented items are the demands fulfilled this period and the items that have
not yet been returned. The state after transshipments is given by

\[ S^E_t \equiv S^E_t(S^R_t, a^E_t, D_t, R_t) = (x^E_t, y^E_t). \]  

(2.12)

The costs of transshipments for a given state \( S^R_t \) and demand \( D_t \) are given by:

\[ C^E(S^R_t, D_t, a^E_t) = c^E \cdot \min(\sum_{i=1}^{n} (x^R_{it} - D_{it})^+, \sum_{i=1}^{n} (x^R_{it} - D_{it})^-), \]  

(2.13)

which is the total number of transshipments multiplied by the unit cost of a trans-shipment. The total number of transshipments is the minimum of excess demand and excess stock. The costs are independent of the transshipment choice \( a^E_t \), because the unit transshipment cost is independent of the library, and we fulfill each possible demand. However, the choice of \( a^E_t \) does influence the state reached after the transshipment and returns.

As example, suppose that \( x^R_t = (1, 1, 1) \) and \( y^R_t = (0, 2, 0) \). The demand in this period is \( D_t = (2, 2, 0) \), so that after demand stock levels are given by \( x^R_t - D_t = (-1, -1, 1) \). There are two transshipment possibilities: \( a^E_t = (1, 0, -1) \) and \( a^E_t = (0, 1, -1) \). If we choose \( a^E_t = (1, 0, -1) \), the unfulfilled demand will be \( m_t = (0, 1, 0) \). When the returns are \( R_t = (0, 1, 0) \), the new stock is given by \( x^E_t = (0, 1, 0) \) and the rented stock is given by \( y^E_t = (2, 2, 0) \).

### 2.3.5 Value Functions and Dynamic Programming Algorithm

In the finite horizon dynamic program, the value functions \( V^R_t(S^E_{t-1}) \) are calculated iteratively by moving from the final period \( T \) to the first. A value function is an iterative function that gives the expected future costs of being in a state at a given period. Each period, the decision that minimizes expected costs for the rest of the horizon is determined for every state. This implies that for making a decision in period \( t \), it suffices to know the values of the states in period \( t + 1 \) to determine an optimal decision. The optimal decisions for the states in the first period define the optimal policy for the chosen time horizon. We assume that repositioning is the first action at time \( t = 1 \).

The value function for repositioning at time \( t \), \( V^R_t(S^E_{t-1}) \), gives the future repositioning costs when the system ended up in the transshipment state \( S^E_{t-1} \) in period \( t - 1 \). These future costs depend on the repositioning decision we make now, \( a^R_t \), and
Chapter 2

the expected costs for the state that we reach after repositioning, $S_t^R$. Assuming that the first items are repositioned in period 1, a period $t$ is a repositioning period if and only if

$$(t - 1) \mod T = 0.$$ 

Let the value of the transshipment state for a demand $D_t$ be $V_t^E(S_t^R, D_t)$. Then the value function of repositioning is given by:

$$V_t^R(S_{t-1}^E) = \min_{a_t^R} \left\{ C^R(S_{t-1}^E, a_t^R) + \mathbb{E}_{D_t} \left[ V_t^E(S_t^R, D_t) \right] \right\}. \quad (2.14)$$

The value function contains immediate repositioning costs, and the expectation over the demand distribution of costs for transshipments in the post-repositioning state $S_t^R = S_t^R(S_{t-1}^E, a_t^R)$. When $t$ is not a repositioning period, we set $a_t^R = (0, \ldots, 0)$.

In period $t$, when the state is $S_t^R$ and the demand is $D_t$, we take the transshipment decision that minimizes the immediate transshipment costs and expected future repositioning costs over the return distribution:

$$V_t^E(S_t^R, D_t) = \min_{a_t^E} \left\{ C^E(S_t^R, D_t) + \mathbb{E}_{R_t} \left[ V_t^R(S_t^E) \right] \right\}. \quad (2.15)$$

Here, $S_t^E = S_t^E(S_t^R, a_t^E, D_t, R_t)$ is the state reached after transshipments and returns. The decision $a_t^E$ influences the state reached in the future, because items are moved to/from different libraries depending on the decision. By making a good transshipment decision, we can reach future states with low costs.

The value functions for the final period have to be initialized before we start. For each state $S_T^R$ and demand $D_T$ in the final period $T$, the value function for transshipments equals the final period transshipment costs. There are no future repositioning costs to take into account, since the time horizon has ended. Any transshipment choice has equal costs, so that there is no minimization involved. The value function of transshipments in the period $T$ is given by:


The final period value function for repositioning is given by evaluating equation (2.14) at time $T$.

The choice of $T$ has influence the first period decisions. In case a low value of
Transshipments and Repositioning in the Standard Single-echelon Rental System 27

$T$ is selected, the decisions will often by myopic. For instance, when there is only one period left, repositioning is often not worthwhile because possible future gains are minimal. A sufficiently long time horizon must therefore be chosen to mitigate the effect of the terminal conditions. The repositioning and transshipment decisions converge within 30 periods, and usually faster.

Below, the steps of the dynamic programming algorithm are summarized. First, the final state value functions are initialized. Then, the algorithm moves back in time and iteratively calculates the value functions until the first period is reached. Since for each period in time the optimal decision for the future is taken, this iterative procedure returns an optimal solution. The running time of the dynamic programming algorithm grows exponentially, because the size of the state space, which is given by $(2^n + K - 1)_K$, grows exponentially.

Step 0: Initialize the final period transshipment and repositioning value functions and set $t = T - 1$.
Step 1: Calculate the value function for transshipments, $V^E_t(S^R_t, D_t)$, for each repositioning state $S^R_t$ and demand $D_t$.
Step 2: Calculate the value function for repositioning $V^R_t(S^E_{t-1})$ for each transshipment state $S^E_{t-1}$, with no repositioning when $t$ is not a repositioning period.
Step 3: If $t > 1$, decrement $t$ and return to step 1. Else, stop.

2.4 Clustering Heuristic

Optimal solutions can be generated with the model from §2.3, however, the size of the solvable instances is limited due to the time-complexity of the algorithm. Therefore, for larger instances, we introduce the cluster method. The cluster method is largely inspired by practice. In practice, libraries often cooperate with other libraries that are located in the vicinity. Transshipments are often handled within the cluster before asking help from another cluster. Therefore, we propose to use our optimal algorithm within the cluster first, and then focus on finding a feasible solution for the full system when transshipments between clusters are required. This gives a good indication of the maximum performance that libraries may attain in the current mode of cooperation.

Let $\Gamma = \{1, \ldots, n\}$ be the set of all libraries, and let clusters $\Gamma_j, j = 1, \ldots, J$, be...
disjoint subsets of $\Gamma$ such that $\bigcup_{j=1}^J \Gamma_j = \Gamma$. For each cluster individually, we define the state space as in §2.3 with the addition of an artificial library that represents the aggregate available and rented items outside the cluster. The artificial library is required to model the exchange of items between clusters. The demand distribution for a cluster’s state space contains the demand within the cluster, and the aggregate demand of the libraries outside the cluster. It is easy to see that each cluster’s state space is smaller than the original state space. To ensure that transshipments are handled within the cluster first, the constraints (2.6)-(2.10) need to be modified. Let $n_j = |\Gamma_j|$ be the number of libraries in cluster $\Gamma_j$, $j = 1, \ldots, J$. The index of the artificial library in cluster $\Gamma_j$ is then $n_j + 1$. Let $D_{n_j+1,t} = \sum_{i \in \Gamma \setminus \Gamma_j} D_{it}$ be the aggregate demand for the artificial library of cluster $\Gamma_j$ at time $t$. The number of items transshipped to the artificial library in response to stock-outs is given by:

$$a_{n_j+1,t}^E = \min \left( \left( \begin{array}{l} \sum_{i=1}^{n_j} x_{it} - \sum_{i=1}^{n_j} D_{it}^+,(x_{n_j+1,t} - D_{n_j+1,t})^- \end{array} \right), \right)$$

$$\left( \begin{array}{l} \sum_{i=1}^{n_j} x_{it} - \sum_{i=1}^{n_j} D_{it}^-,(x_{n_j+1,t} - D_{n_j+1,t})^+ \end{array} \right),$$

i.e., we only transship to/from the artificial library when there is a stock-out either inside or outside the cluster. The sum of the mutations within the cluster then adds up to $\sum_{i=1}^{n_j} a_{it}^E = -a_{n_j+1,t}^E$, and the change for the other restrictions is straightforward. We can run our optimal algorithm from §2.3 to find the solutions for each of the clusters.

The solutions for the clusters need to be translated to a solution for the full library system, $\Gamma$. For the transshipments, the following is proposed. Let $S_t^R$ be the state in the original state space after repositioning. Let $D_t$ be the demand. For cluster $\Gamma_j$, $j = 1, \ldots, J$, the corresponding cluster states and demands are given by $S_t^{\Gamma_j,R}$ and $D_t^{\Gamma_j}$, which can be determined straightforwardly. Let the optimal within-cluster transshipment decision for cluster $\Gamma_j$ be given by $a_t^{\Gamma_j,E}$. The following procedure describes a heuristic approach to arrive at a solution for the system as a whole:
Step 1: Construct the transshipment decision:
\[
a_t^E = (a_{1t}^{\Gamma_1,E}, \ldots, a_{nt}^{\Gamma_1,E}, a_{1t}^{\Gamma_2,E}, \ldots, a_{nt}^{\Gamma_2,E}, \ldots, a_{1t}^{\Gamma_J,E}, \ldots, a_{nt}^{\Gamma_J,E}).
\]
If \( \sum_{i=1}^n a_{it}^E = 0 \), then \( a_t^E \) is feasible. Stop. Else go to step 2.

Step 2: Calculate the total required number of transshipments \( A \), i.e.,
\[
A = \min(\sum_{i=1}^n (x_{it} - D_{it})^+, \sum_{i=1}^n (x_{it} - D_{it})^-).
\]
If \( \sum_{i=1}^n (a_{it}^E)^+ < A \) then iteratively transship \( A - \sum_{i=1}^n (a_{it}^E)^+ \) items in total to libraries \( i \) with \( (x_{it}^R - D_{it} + a_{it}^E) < 0 \). If \( \sum_{i=1}^n (a_{it}^E)^- < A \) then iteratively transship \( A - \sum_{i=1}^n (a_{it}^E)^- \) items in total from libraries \( i \) with \( (x_{it}^R - D_{it} + a_{it}^E) > 0 \). Libraries with the largest decrease in expected stock-out costs should receive the transshipments, and libraries with the smallest increase in expected stock-out costs should send the transshipments (see §2.5).

Note that for two clusters, step 1 always gives a feasible transshipment. This holds, because the total number of incoming transshipments in cluster \( \Gamma_1 \) equals the number of outgoing transshipments in cluster \( \Gamma_2 \). When there are three or more clusters this does not hold in general. As an example, consider the case where cluster \( \Gamma_1 \) has 1 item left, and cluster \( \Gamma_2 \) and \( \Gamma_3 \) both still demand 1 item. Step 2 makes these infeasible transshipment decisions feasible by heuristically selecting receiving and sending locations.

For repositioning there are two options. The first option is to allow full repositioning. We take the heuristic transshipments as given, and then run our optimal algorithm for the full state space to find the optimal repositioning decisions. For very large state spaces, this may still cause tractability issues in which case the second option is to construct a solution from the within-cluster solutions. Feasibility is assured by greedily adding and subtracting stock movements. In the results section, we only look at the first option, so that we can better identify the cost of restricting transshipments within the clusters.
2.5 Expected Shortage Reduction Heuristic

The optimal algorithm is only capable of addressing instances up to 4-5 libraries due to the rapid growth of the state space and the number of transitions. The cluster heuristic can handle larger instances, but still suffers from increase in the size of the state space through the number of items. Moreover, this clustering typically takes place within a municipality, in which there are sometimes more than 4 libraries. For example, the city of Groningen has 7 libraries. For a whole region, there may even be up to 100 libraries. In the province of Groningen, the Netherlands, for instance, there are 61 libraries cooperating. The optimal and cluster method cannot deal with instances of this size. Therefore, for instances with a very large number of libraries and items, the expected shortage reduction (ESR) heuristic is presented below. This heuristic aims at reducing the expected shortage costs in the short term. Since a main characteristic of rental systems is the return of products, the heuristic specifically takes returns into account. Other transshipment rules from the literature are often not suitable in case of returns. The transshipment rule in Axsäter (2003), for instance, cannot be applied for our problem, since it does not account for fixed stock and returns.

2.5.1 Transshipments in the ESR Heuristic

If at any point in time there is more demand than supply, a choice has to be made regarding which demand will be fulfilled and which not. Similarly, if there is more supply than demand, a choice has to be made from where to transship. Our optimal algorithm makes this choice in such a way that future expected costs are minimized. The heuristic we present here follows the same basic principle, but instead of looking forward for all future periods, it will look ahead by only two periods. Once demand is observed, the total number of required transshipments is known. The ESR heuristic transships from the library with the smallest increase in two-period-ahead expected shortages after the transshipment. Items are transshipped to the library that gives the largest decrease in expected two-period-ahead shortages. This will lead to a minimum amount of expected shortages two periods from now.

Let $D_{i0}$, $D_{i1}$, and $D_{i2}$ denote the demand at library $i$ now, 1 period from now, and 2 periods from now. Let $R_{i0}$ and $R_{i1}$ be the returns at library $i$ now and in period 1. Let the pre-demand stock levels 1 and 2 periods from now be given by
Transshipments and Repositioning in the Standard Single-echelon Rental System

\[ x_{i1} = (x_{i0} - D_{i0} + a_i^E)^+ + R_{i0} \quad \text{and} \quad x_{i2} = (x_{i1} - D_{i1})^+ + R_{i1} \]. The expected shortage for two periods ahead for library \( i \) is given by:

\[
f(i) = \mathbb{E}_{R_{i0},D_{i1}} [(x_{i1} - D_{i1})^- + \mathbb{E}_{R_{i1},D_{i2}} [(x_{i2} - D_{i2})^- | R_{i0}, D_{i1}^1] ,
\]

where no further interaction with other libraries is assumed for the future periods. Let \( f^-(i) \) and \( f^+(i) \) denote the expected shortage when one item is removed and one is added, respectively. The procedure below describes the heuristic formally:

**Step 0:** For the observed state and demand calculate the number of transshipments \( A \):

\[
A = \min \left( \sum_{i=1}^{n} (x_{i0} - D_{i0})^+, \sum_{i=1}^{n} (x_{i0} - D_{i0})^- \right).
\]

The initial transshipment decision is \( a^E = (0, \ldots, 0) \). If \( A = 0 \), stop. Else, set \( j = 1 \) and go to step 1.

**Step 1:** The current vector of unfulfilled demand is given by: \( m_0 = (x_0 - D_0 + a^E)^- \). Let \( I^- \) be the set of libraries from which an item can be transshipped and \( I^+ \) the libraries that require a transshipment, i.e.: \( I^- = \{ i : x_{i0} - D_{i0} + a_i^E > 0 \} \), and \( I^+ = \{ i : x_{i0} - D_{i0} + a_i^E < 0 \} \).

**Step 2:** Calculate the expected shortages currently, when an item is removed, and when an item is added. That is, calculate \( f(i) \) for \( i = 1, \ldots, n \), \( f^-(i) \) for \( i \in I^- \), and \( f^+(i) \) for \( i \in I^+ \).

**Step 3:** Now let

\[
k = \arg \min_{i \in I^-} (f^-(i) - f(i)), \text{ and } \ell = \arg \max_{i \in I^+} (f(i) - f^+(i)).
\]

In case of ties, \( k \) and \( \ell \) are assigned the lowest index. Let the new decision be: \( a^E := a^E - e_k + e_\ell \), with \( e_i \) the Euclidean vector.

**Step 4:** Set \( j := j + 1 \). If \( j < A \), go to step 1, else stop.

### 2.5.2 Repositioning in the ESR Heuristic

For repositioning we propose a similar approach as for transshipments. For each library, the expected shortage costs for the upcoming two periods are calculated for
three situations: the situation currently, when one unit is removed, and when one unit is added. The library that has the largest decrease in expected future costs when adding a unit is the candidate for transshipping an item to. The library that has the smallest increase in expected costs when a unit is removed is the candidate to transship an item from. If the total decrease of expected costs exceeds the unit repositioning cost, in principle we reposition the item between the two candidates. However, we cancel the repositioning action when it leads to a deterioration of the expected shortage costs in the current period. Then it seems better to postpone the action. Finally, we check whether there are cost-efficient improvements for the immediate period.

Let the two-period-ahead expected shortage after the repositioning phase be given by

$$g_2(i) = \mathbb{E}_{D_{i0}} \left[ (x_{i1})^- + \mathbb{E}_{R_{i0},D_{i1}} \left[ (x_{i1})^+ + R_{i0} - D_{i1} \right] D_{i0} \right],$$

with $x_{i1} = x_{i0} - D_{i0} + a_i^R$. The one-period-ahead expected shortage is given by

$$g_1(i) = \mathbb{E}_{D_{i0}} \left[ (x_{i1})^- \right].$$

Again, we assume that there is no repositioning and transshipping in future periods. Let $g_1^-(i)$ and $g_2^-(i)$ be the expected shortages when a unit is removed, and $g_1^+(i)$ and $g_2^+(i)$ when a unit is added to library $i$. The heuristic is as follows:

**Step 0:** Let the initial repositioning decision be $a^R = (0, \ldots, 0)$.

**Step 1:** Let $I^-$ be the set of libraries that can spare a unit (libraries with positive stock). Calculate the one- and two-period-ahead expected shortages: $g_1(i), g_1^+(i), g_1^-(i), g_2(i), g_2^+(i)$, and $g_2^-(i)$ for $i = 1, \ldots, n$

**Step 2:** Determine the library from $I^-$ with lowest increase in expected shortage when a unit is removed, and the library that has the highest decrease when a unit is added:

$$k = \arg \min_{i \in I^-} [g_2^-(i) - g_2(i)], \quad \text{and} \quad \ell = \arg \max_{i = 1, \ldots, n} [g_2(i) - g_2^+(i)].$$

If

$$[g_2(\ell) - g_2^+(\ell)] - [g_2^-(k) - g_2(k)] > c^R/c^E$$
and
\[ g_1(\ell) - g_1^+(\ell) \geq g_1^-(k) - g_1(k) \]
then \( a^R := a^R - e_k + e_\ell \) and return to step 1. Else, continue to step 3.

**Step 3:** Calculate the one-period-ahead expected shortages, \( g_1(i), g_1^+(i), \) and \( g_1^-(i) \) for \( i = 1, \ldots, n \).

**Step 4:** Determine:

\[ k = \arg \min_{i \in I^-} [g_1^-(i) - g_1(i)], \quad \text{and} \quad \ell = \arg \max_{i=1,\ldots,n} [g_1(i) - g_1^+(i)]. \]

If
\[ [g_1(\ell) - g_1^+(\ell)] - [g_1^-(k) - g_1(k)] > c_R/c_E \]
then \( a^R := a^R - e_k + e_\ell \) and return to step 3. Else, stop.

### 2.5.3 Heuristics from Practice

In practice, libraries typically return items to the library they originated from. The reason for this is that libraries prefer to maintain their own collection. This fixed ownership policy is straightforward to implement: in the state space, we count a transshipped item as a rented item for the original library. The costs for a transshipment are simply the unit transshipment cost plus the unit repositioning cost, \( c_E + c_R \).

A returned item in the fixed ownership policy is assumed to be sent back to its owner library immediately. Note that the state space for the fixed ownership case is smaller than the state space of the original model, since the sum of available and rented items remains constant for each library.

An alternative policy, that is gaining interest in practice, is the floating policy (see, e.g., [Van der Noordaa 2011](#)). Items float through the system via transshipments and remain at the libraries to which they have been transshipped. There is no repositioning, so this is equivalent to setting \( a_t^R = (0, \ldots, 0) \) for any state and time.

For transshipments in practice often simple procedures based on employees’ habits are used. The alphabetical order of the libraries is often the main selection criterion for transshipment. When an item is requested, librarians retrieve a list from the computer system with libraries that have the item in stock. Since the transshipment
costs of all options are identical for the requesting library, the librarian typically tends to simply select the first from the list. From the decentralized view of the requesting library this is reasonable, but from a centralized view this may lead to very low stock at libraries low in the alphabet. We will call this the alphabet transshipment policy.

### 2.6 Experimental Results

The results for the optimal algorithm and the heuristics are discussed in this section. Optimal repositioning is discussed first. Then the costs for the ESR heuristic are calculated in a simulation experiment and compared with the optimal policy and policies from practice. Finally, costs are compared for large sized instances in a setting resembling practice. The experiments are run on a i3-2120 CPU (3.3 GHz) with 3.16 GB working memory.

#### 2.6.1 Example of Repositioning in the Optimal Policy

In Table 2.1 we show the results of repositioning in an example for the $n = 3$ case with varying library probabilities, cost-ratios, and rented items. Let the library demand probabilities for 0, 1, or 2 items for a high (H), medium (M) and low (L) demand library be given by: $H = (0.4, 0.48, 0.12)$, $M = (0.6, 0.32, 0.08)$, and $L = (0.8, 0.18, 0.02)$. The cost-ratio, $c^E/c^R$, denotes the relative cost of transshipping compared to repositioning. In this example, the return probability is $r = 1$, which means items return one period after they have been rented. Moreover, $\bar{T} = 1$, so that we reposition every period. This example could for instance model the case of a DVD rental company that allows DVDs to be rented only for a single day. Since there are close to 300 states for the $n = 3$ case, we only show the results for several states. In these representative states, the vector of available items is given by $x_t = (0, 1, 3)$. The rightmost column shows the available items after repositioning. Similar results hold for other states.

The results in Table 2.1 suggest that the optimal repositioning decision in the example depends on all three factors. For a low cost-ratio, when repositioning is relatively expensive, the decision sometimes depends on the rented items. We take the risk of having a stock-out in the immediate period, so that after the item has returned, the system is in balance. When the cost-ratio is higher, repositioning becomes so cheap that we only aim to reduce the expected costs for the immediate period: the rented items have low influence. The library probabilities also appear to play a vital
Table 2.1: Optimal repositioning in a representative state under varied conditions

<table>
<thead>
<tr>
<th>Available items</th>
<th>Probabilities</th>
<th>Cost-ratio</th>
<th>Rented items</th>
<th>After</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0, 1, 3)</td>
<td>L,L,L</td>
<td>1.5</td>
<td>(1, 1, 0)</td>
<td>(0, 1, 3)</td>
</tr>
<tr>
<td>(0, 1, 3)</td>
<td>L,L,L</td>
<td>1.5</td>
<td>(0, 1, 1)</td>
<td>(1, 1, 2)</td>
</tr>
<tr>
<td>(0, 1, 3)</td>
<td>L,L,L</td>
<td>3</td>
<td>(1, 1, 0)</td>
<td>(1, 1, 2)</td>
</tr>
<tr>
<td>(0, 1, 3)</td>
<td>L,L,L</td>
<td>3</td>
<td>(0, 1, 1)</td>
<td>(1, 1, 2)</td>
</tr>
<tr>
<td>(0, 1, 3)</td>
<td>L,L,L</td>
<td>20</td>
<td>(1, 1, 0)</td>
<td>(1, 1, 2)</td>
</tr>
<tr>
<td>(0, 1, 3)</td>
<td>L,L,L</td>
<td>20</td>
<td>(0, 1, 1)</td>
<td>(1, 1, 2)</td>
</tr>
<tr>
<td>(0, 1, 3)</td>
<td>H,M,L</td>
<td>1.5</td>
<td>(1, 1, 0)</td>
<td>(1, 1, 2)</td>
</tr>
<tr>
<td>(0, 1, 3)</td>
<td>H,M,L</td>
<td>1.5</td>
<td>(0, 1, 1)</td>
<td>(2, 1, 1)</td>
</tr>
<tr>
<td>(0, 1, 3)</td>
<td>H,M,L</td>
<td>3</td>
<td>(1, 1, 0)</td>
<td>(1, 1, 2)</td>
</tr>
<tr>
<td>(0, 1, 3)</td>
<td>H,M,L</td>
<td>3</td>
<td>(0, 1, 1)</td>
<td>(2, 1, 1)</td>
</tr>
<tr>
<td>(0, 1, 3)</td>
<td>H,M,L</td>
<td>20</td>
<td>(1, 1, 0)</td>
<td>(2, 1, 1)</td>
</tr>
<tr>
<td>(0, 1, 3)</td>
<td>H,M,L</td>
<td>20</td>
<td>(0, 1, 1)</td>
<td>(2, 1, 1)</td>
</tr>
</tbody>
</table>

role. A high demand library typically receives more items after repositioning than a low demand library. The optimal decision is the result of the interplay of the three factors. Typical decisions aim for reduction of immediate transshipment costs and for long run stability.

2.6.2 Timing of Repositioning and its Influence on Costs

The timing of repositioning is an important aspect for practice. In our problem formulation, it is evident that lowest costs are obtained by repositioning every period. The libraries in practice, however, may be unable to facilitate repositioning every period. For them, it is typically most practical to reposition on a fixed day in the week or month. In order to study how this timing influences costs, we therefore conduct an experiment in which the time varies between successive moments of repositioning. Figure 2.2 displays the percentage cost increase in case of varying time between repositioning \( \bar{T} \). As a measure for the costs, we take the average of the first period repositioning value functions over all possible states in an experiment with a time horizon of \( T = 120 \) periods. The percentages in Figure 2.2 denote the relative difference in costs compared to repositioning every single period. Three different situations are compared. In the base situation, each library has a demand distribution \((p_{i0}, p_{i1}, p_{i2}) = (0.8, 0.18, 0.02)\), the return probability is \( r = 0.25 \), and transshipment cost \( c^E = 3 \). The base situation shows that costs increase as the time between repo-
positioning increases. When unit transshipment costs are higher, i.e., \( c^E = 5 \), and when the return probability increases to \( r = 1 \), the increase in costs is more severe. The loss due to repositioning infrequently becomes almost 20%.

![Influence of the timing of repositioning on costs](image)

**Figure 2.2:** Cost increase for repositioning every \( \bar{T} \) periods rather than every period

### 2.6.3 Performance Comparison of the Optimal Algorithm and the Heuristics

In a simulation experiment, the effectiveness of the heuristics is compared to our optimal algorithm and to policies from practice. Each experiment in Table 2.2 gives the average cost difference for 50 configurations. In each configuration, the parameters \( n, K \), cost-ratio \( \frac{c^E}{c^R} \), and return probability \( r \) are constant, but library probabilities are variable. The return rates \( r = 0.05 \) and \( r = 0.24 \) correspond to the daily and weekly return rate for library items, which we have estimated from a dataset with 60,000 return transactions. For each library, the probability of zero demands is sampled from a \( U(0.6, 0.98) \) distribution. The probability of 1 and 2 demands are given by
0.9 and 0.1 times the probability of a nonzero demand. For each configuration, we simulated the costs of the policies 10,000 times with a time horizon of 1000 working days, in which we reposition every 5 working days. Common random numbers are used for the starting state and the observed demands across policies. An exception is made for the fixed ownership policy: its starting state is sampled independently from the other policies due to the state space for this policy being unequal to the full state space.

In Table 2.2 the heuristics are indexed by two terms: the first term refers to the method used for transshipments and the second for repositioning. ESR/Optimal, for example, means that transshipments are calculated with the ESR heuristic and these are combined with the repositioning decisions from the optimal algorithm. In the fixed ownership policy, each library has to be assigned an initial stock, since the sum of available and rented items per library remains constant. Each library has been assigned two items, since in practice libraries hardly ever have more than two items. For the cluster method the $n = 3$ case has not been simulated, since there is only one cluster of size 3 and hence the solution equals the optimal solution. For the $n = 4$ case, each cluster has been assigned two libraries.

The effectiveness of our proposed transshipment heuristics can be observed in the columns ESR/Optimal and Cluster/Optimal. The ESR transshipments introduce an average cost increase of at most 0.35% compared to optimal, while the cluster transshipments lead to increases between 1% and 12%. The ESR transshipment heuristic is close to optimal and performs robustly in each experiment. It seems that the ESR heuristic almost always picks the right transshipment candidate. Contrary to the ESR heuristic, the cluster heuristic often selects a suboptimal candidate for transshipment. This is due to restricting transshipments within the cluster: the best transshipment candidate is not necessarily located in the cluster. It is evident that the ESR heuristic is preferable for the instances analyzed. However, in situations with a different cost structure, where within-cluster transshipments are cheaper than between-cluster transshipments, the cluster method is likely to show a relative improvement. This may occur when within the cluster there is a cheap transportation system, while for longer distances mail carriers are used.

The columns with ESR/ESR and ESR/Floating show how costs increase with different policies for repositioning. In all experiments, ESR repositioning has lower average costs than not repositioning (floating). The ESR heuristic performs close to
Table 2.2: Percentage cost difference with the optimal solutions for several heuristic approaches

<table>
<thead>
<tr>
<th>Experiment</th>
<th>$n$</th>
<th>$K$</th>
<th>$c_{ik}^E$</th>
<th>$r$</th>
<th>ESR/Optimal</th>
<th>Cluster/Optimal</th>
<th>ESR/ESR</th>
<th>ESR/Floating</th>
<th>Alphabet/Fixed Ownership</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>0.05</td>
<td>0.02</td>
<td>-</td>
<td>1.04</td>
<td>3.28</td>
<td>71.70</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>0.24</td>
<td>0.31</td>
<td>-</td>
<td>4.89</td>
<td>7.96</td>
<td>70.76</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>1</td>
<td>0.18</td>
<td>-</td>
<td>8.50</td>
<td>9.90</td>
<td>31.48</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>6</td>
<td>10</td>
<td>0.05</td>
<td>0.03</td>
<td>-</td>
<td>0.49</td>
<td>8.93</td>
<td>40.55</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>6</td>
<td>10</td>
<td>0.24</td>
<td>0.03</td>
<td>-</td>
<td>3.99</td>
<td>20.70</td>
<td>53.52</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>6</td>
<td>10</td>
<td>1</td>
<td>0.00</td>
<td>-</td>
<td>17.20</td>
<td>30.45</td>
<td>35.61</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>8</td>
<td>3</td>
<td>0.05</td>
<td>0.03</td>
<td>1.50</td>
<td>0.53</td>
<td>1.94</td>
<td>59.60</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>8</td>
<td>3</td>
<td>0.24</td>
<td>0.35</td>
<td>5.23</td>
<td>2.83</td>
<td>6.02</td>
<td>70.74</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>8</td>
<td>3</td>
<td>1</td>
<td>0.35</td>
<td>11.93</td>
<td>6.31</td>
<td>7.88</td>
<td>49.65</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>8</td>
<td>10</td>
<td>0.05</td>
<td>0.02</td>
<td>1.09</td>
<td>0.20</td>
<td>6.74</td>
<td>36.22</td>
</tr>
<tr>
<td>11</td>
<td>4</td>
<td>8</td>
<td>10</td>
<td>0.24</td>
<td>0.03</td>
<td>3.52</td>
<td>2.71</td>
<td>17.17</td>
<td>46.96</td>
</tr>
<tr>
<td>12</td>
<td>4</td>
<td>8</td>
<td>10</td>
<td>1</td>
<td>0.07</td>
<td>5.47</td>
<td>13.79</td>
<td>25.04</td>
<td>32.60</td>
</tr>
</tbody>
</table>

optimal when the return probability is $r = 0.05$ and within 5% from optimality when $r = 0.24$. These return probabilities correspond to the actual data of transactions, indicating that the ESR heuristic performs well in practical instances. In the case with $r = 1$, the ESR heuristic can lead to increased costs of 17% on average. This increase is mostly caused by configurations where demand is low. In these situations, the change in two-period expected shortages is so small that the heuristic cannot identify the repositioning decisions that are optimal for the long run. Total costs for a scenario with low demand are fairly minimal, however, so that the difference in absolute cost is limited. The floating policy performs reasonably well when transshipments are cheap compared to repositioning, but performs worse when transshipments are expensive.

The method most commonly used in practice, the fixed ownership policy with alphabet transshipments, has the worst performance in the simulation. Because the policy holds on to the initial allocation of items, items are moved more frequently.
than with the other policies. There are always two stock movements corresponding to a transshipment, which gives rise to substantially higher costs than necessary. Moreover, always transshipping from the first library with stock often excludes the best transshipment candidate. The results for the fixed ownership policy are even slightly optimistic, since we allow for sending back items every working day, rather than every five working days.

2.6.4 Comparison of the ESR Heuristic with Policies from Practice for a Larger Number of Libraries

The computation times and memory usage of the optimal algorithm and the cluster heuristic grow quickly when the instance-size increases. Since practical instances exist with up to 70 libraries, we would like to investigate the quality of the heuristics in instances of that size. Therefore, we compare the ESR heuristic to the various policies used in practice for large \( n \) in a simulation experiment. The returns in this experiment are drawn from a dataset with actual returns. In this way, we can observe whether or not it is appropriate to assume a fixed return probability. Note that for the case with \( n = 100 \) libraries and \( K = 200 \) items, a desktop computer is capable of calculating the ESR transshipment decisions for approximately 55,000 item types per hour, and the ESR repositioning decisions for about 85,000 item types per hour. This is reasonable for practical purposes, since the decisions are taken on a daily/weekly basis.

Table 2.3 shows the average percentage cost improvement over fixed ownership policy for three heuristics in a case with varying \( n \). The fixed ownership policy is combined with alphabet transshipments. The terms A/FL and ESR/FL are shorthand for floating policy combined with alphabet and ESR transshipments, respectively. The average is taken over 50 configurations, each with a random number of starting items in the range \( n, n+1, \ldots, 2n \). Each configuration is simulated 100 times with a time horizon of 100 weeks. The transshipment costs for a configuration are uniformly \( U(3,10) \) distributed and libraries have the same type of demand probabilities as for the experiment in Table 2.2. The choice of \( r \) in the ESR heuristic is 0.24, which corresponds to the return probability fitted from a dataset with book transactions. The experiment is run with a discretization step of a week, and repositioning also takes place on a weekly basis.

Table 2.3 shows that all three policies have lower costs than the fixed ownership policy. The ESR/ESR policy consistently gives the largest improvement over practice,
Table 2.3: Average percentage cost improvement of three heuristics over the fixed ownership policy.

<table>
<thead>
<tr>
<th>n</th>
<th>ESR/ESR</th>
<th>A/FL</th>
<th>ESR/FL</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>31.81</td>
<td>12.17</td>
<td>16.17</td>
</tr>
<tr>
<td>5</td>
<td>33.25</td>
<td>10.74</td>
<td>19.75</td>
</tr>
<tr>
<td>10</td>
<td>36.23</td>
<td>7.16</td>
<td>23.86</td>
</tr>
<tr>
<td>50</td>
<td>36.85</td>
<td>3.50</td>
<td>25.10</td>
</tr>
<tr>
<td>100</td>
<td>36.79</td>
<td>4.68</td>
<td>26.94</td>
</tr>
</tbody>
</table>

and this relative gap seems to be robust in the number of libraries. Especially interesting is the development of costs for the floating policy with different transshipment rules. While the relative gain of not repositioning decreases in \( n \) with alphabet transshipments, it increases with ESR transshipments. This indicates that it is important to have a balanced transshipment policy when the number of libraries increases. Otherwise, items are too frequently stored in libraries in which they are not required. The results indicate that a dynamically moving collection driven by demand may improve availability and lower costs. The primary cost gain is that items do not always have to be moved twice when a transshipment is required. The experiment also shows that, even with the assumption of a fixed return probability, the ESR heuristic remains viable in a setting in which returns follow the actual frequency distribution.

### 2.7 Conclusion

In this chapter, a novel problem for library control was presented and solved by stochastic dynamic programming and a new effective heuristic. In contrast to previous library literature, we considered the proactive repositioning of library items. Optimal repositioning and transshipment policies are derived from the model, which yield important insights for practice. The main message of the results is that libraries can achieve lower costs at the same service level by coordinating transshipments cooperatively. By addressing transshipments correctly and by proactively repositioning items in the system, costs can be reduced. The results indicate that the fixed ownership policy from practice, in which items are always returned to their home library, is hard to defend from a cost perspective. Floating library collections driven by demand seem to be preferable. Clustering of libraries, as is often done in practice, appears to
Transshipments and Repositioning in the Standard Single-echelon Rental System

have a negative impact on efficiency in case the handling costs for transshipments are the main driver for the costs. The ESR heuristic that we formulated seems to be a promising tool for decision making in library systems with online demand.