European Journal of Operational Research 228 (2013) 367–380

Contents lists available at SciVerse ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Production, Manufacturing and Logistics

Dynamic demand fulfillment in spare parts networks with multiple customer classes

H.G.H. Tiemessen [a,*], M. Fleischmann [b], G.J. van Houtum [c], J.A.E.E. van Nunen [d], E. Pratsini [a]

[a] Department of Mathematical and Computational Sciences, IBM Research, Zurich Research Laboratory, 8803 Rüschlikon, Switzerland
[b] Business School, University of Mannheim, P.O. Box 10 34 62, 68131 Mannheim, Germany
[c] School of Industrial Engineering, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands
[d] Rotterdam School of Management, Erasmus University, P.O. Box 1738, 3000 DR Rotterdam, The Netherlands

ARTICLE INFO

Article history:
Received 26 January 2012
Accepted 20 January 2013
Available online 31 January 2013

Keywords:
Inventory
Spare parts
Dynamic demand fulfillment
Multiple customer classes

ABSTRACT

We study real-time demand fulfillment for networks consisting of multiple local warehouses, where spare parts of expensive technical systems are kept on stock for customers with different service contracts. Each service contract specifies a maximum response time in case of a failure and hourly penalty costs for contract violations. Part requests can be fulfilled from multiple local warehouses via a regular delivery, or from an external source with ample capacity via an expensive emergency delivery. The objective is to minimize delivery cost and penalty cost by smartly allocating items from the available network stock to arriving part requests. We propose a dynamic allocation rule that belongs to the class of one-step lookahead policies. To approximate the optimal relative cost, we develop an iterative calculation scheme that estimates the expected total cost over an infinite time horizon, assuming that future demands are fulfilled according to a simple static allocation rule. In a series of numerical experiments, we compare our dynamic allocation rule with the optimal allocation rule, and a simple but widely used static allocation rule. We show that the dynamic allocation rule has a small optimality gap and that it achieves an average cost reduction of 7.9% compared to the static allocation rule on a large test bed containing problem instances of real-life size.

1. Introduction

We consider demand fulfillment for networks consisting of multiple local warehouses, where spare parts of expensive technical systems are kept on stock. Downtime costs (like opportunity cost in case of lost production, liability cost, or loss of goodwill) are usually high, and can easily run into thousands of euros per hour. Therefore, it is important that the availability of the systems is high and that down-situations are recovered quickly. Original Equipment Manufacturers (OEMs) of high-tech equipment usually sell their equipment with a variety of service contracts at different prices. Typically, these service contracts commit to maintenance service within 2 hours, 4 hours, 8 hours, or the next, or second next business day. Due to strong fluctuations in spare parts demand and strict service deadlines, spare parts logistics execution must be responsive. This is achieved by means of fast call-handling, accurate (remote-) problem diagnosis, 24 hours operations, fast transportation modes, and stocking locations at close distance to the customer site.

OEMs often serve customers with different service deadlines from the same network. Because of the pooling effect, this requires (much) lower investments in inventory than operating separate networks per contract type. To serve customers with premium contracts in time, OEMs must establish dense same day networks of stocking locations. Consequently, many customers can be served within the service deadline from multiple stocking locations.

A major challenge for the OEM is to minimize inventory holding cost, replenishment cost and fulfillment cost while providing the promised service to its customers. OEMs can simultaneously influence cost and customer service in two possible ways: (i) through calculating appropriate base stock levels for all warehouses, and (ii) through determining an appropriate allocation rule for selecting the warehouse that is used to fulfill a real-time spare parts demand.

Whereas the calculation of base stock levels is a tactical planning problem where decisions are usually taken every 3–6 months, the calculation of a stock allocation rule is an operational planning problem where usually many decisions must be taken each day. As for many hierarchical planning problems in Supply Chain Management, it is not possible to solve these two problems simulta-

* Corresponding author. Tel.: +41 41 534 29 59; fax: +41 44 724 89 64.
E-mail addresses: hti@zurich.ibm.com (H.G.H. Tiemessen), MFleischmann@bwl.uni-mannheim.de (M. Fleischmann), g.j.v.houtum@tue.nl (G.J. van Houtum), pra@zurich.ibm.com (E. Pratsini).

0377-2217/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.ejor.2013.01.042

neously. In this study we focus on the operational planning problem and assume that the base stock levels are given. Our aim is to develop an allocation rule that performs well for *arbitrary* base stock level vectors. Such an allocation rule can serve as a plug-in for methods that calculate appropriate base stock levels. In Section 6, we discuss methods for calculating base stock levels that anticipate our proposed allocation rule.

In practice, we see that many OEMs have implemented a simple static allocation rule that fulfills a real-time spare parts demand from the closest warehouse with stock at hand. See Reijnen et al. (2009) and Kranenburg and Van Houtum (2009) for case studies at OEMs of high tech equipment. This allocation rule is popular because it only requires static time/distance information and the set of warehouses with stock at hand. One drawback of this rule is that it does not use some potentially valuable information for making smart allocation decisions: real-time stock level information, and demand forecast information. Another drawback of this rule is that it does not consider customer base heterogeneity. As a result, each customer, despite generating different revenues to the OEM, receives a similar treatment in terms of customer service. This is unattractive for the OEM for two reasons: First, customers who bought expensive high-end contracts might consider this unfair, and this might disturb the customer relationship. Second, high base stock levels are needed to provide the promised service to customers with tight service deadlines. It is thus attractive to differentiate between the customers based on their service contract. Customer differentiation can be realized using critical levels. For single location models this is often the only choice. In networks however, customer differentiation can also be realized through dynamic allocation. In this paper we explore this direction.

Our goal in this paper is to investigate the benefits of using real-time stock level information and demand forecast information for real-time demand management in spare parts inventory networks. We summarize our problem setting as follows: We consider a single item, single echelon, multi location inventory network where spare parts are kept on stock for customers with different service contracts. Each service contract specifies a maximum response time in case of a failure, and contract violations are penalized. The objective is to minimize the sum of the average annual delivery cost and the average annual penalty cost by smartly allocating items from the available network stock to arriving part requests. What makes this task challenging is that the choice to fulfill a part request from a particular source location does not only cause a certain direct cost, but also impacts the opportunities to fulfill demands in the near future (at least until the allocated item is replenished again). We present an average cost Markov Decision Process (MDP) formulation of this problem, which enables us to compute the optimal allocation rule for small problem instances. To handle problem instances of real-life size, we develop a one-step lookahead policy where we approximate the optimal relative cost with an estimate of the relative cost under a simple static allocation rule. In a series of numerical experiments, we compare the performance of the proposed allocation rule with the optimal allocation rule and two benchmark allocation rules.

To summarize, the paper makes the following contributions:

- We develop a dynamic allocation rule that is applicable to problem instances of real-life size. This allocation rule is a one-step lookahead policy that takes into account base stock levels, actual stock levels, and demand forecast information. We show that the optimality gap of the proposed allocation rule is usually small (i.e. average gap is less than 2%).
- We characterize our dynamic allocation rule by comparing its allocation decisions to the allocation decisions of a simple static allocation rule. We show that the dynamic allocation rule achieves considerable cost savings by deviating from the simple static allocation rule in a relatively small number of situations. In particular, we show that the dynamic allocation rule is more reluctant to take away the last item at a local warehouse and less reluctant to use emergency deliveries from the central warehouse.
- We show that dynamic allocation leads to significant cost savings compared to a simple but widely used static allocation rule by numerical experiments on a test bed that is inspired by IBM's spare parts network in Europe. We illustrate the impact of key problem characteristics on the potential benefits.

The paper is structured as follows. We start with a literature review and position our research in Section 2. In Section 3, we describe our model and discuss important assumptions. In Section 4, we introduce our new dynamic allocation rule. Section 5 presents a numerical study that compares the proposed dynamic allocation rule with the optimal allocation rule, and two benchmark allocation rules. Finally, in Section 6, we summarize our results and draw conclusions.

2. Literature review

Our work contributes to a rich literature on spare parts inventory models. The literature that is most related to the work in this paper, comes from three streams.

In the first stream of literature, inventory models with fixed lateral transshipment rules are studied. In this stream, demand occurring at a local warehouse with no stock at hand can be fulfilled via a stock transfer from another local warehouse. In literature, models are developed for evaluating system performance, either exactly or approximately. An important contribution in this stream is made by Axsäter (1990) who studied a two echelon model with one central warehouse and a number of local warehouses. In his model demand is fulfilled from the local warehouse whenever possible. If no items are available, an item is sent from a randomly chosen local warehouse with stock at hand. Demand rates observed by each warehouse are approximated by assuming that all demand streams are Poisson. Based on the resulting set of equations, he provides an iterative algorithm to obtain steady state probabilities. In the spirit of Axsäter (1990), Kukreja et al. (2001), Wong et al. (2005), Kutanoglu (2008), Kranenburg and Van Houtum (2009), and Reijnen et al. (2009) develop different models for evaluating and optimizing system performance under given allocation rules. All of these models assume static allocation rules whereas we focus on dynamic allocation rules.

The second stream of literature also studies inventory models with lateral transshipments, but the lateral transshipment rule is now subject to optimization. An important paper in this stream is Axsäter (2003). The author considers a backordering model where a particular local warehouse with no stock at hand receives a customer demand. The task is to select a warehouse to fulfill the demand. The impact of all possible sourcing decisions on the total cost is approximated by considering the direct cost and the additional future cost associated with a temporary reduction of the stock level at the source location. Future cost is calculated under the assumption that no lateral transshipments will take place. Minner et al. (2003) consider a similar model in a retail environment with lost sales. Another interesting contribution in this stream is made by Wijk et al. (2009) who provide an exact analysis for a two location setting with given base stock policies and exponential lead times. What differentiates our work from this stream, is that we consider inventory networks where part requests can be fulfilled directly (i.e. without lateral transshipment) from multiple local warehouses in the network, and that we support multiple customer classes. For a recent and comprehensive literature review on lateral transshipments we refer to Paterson et al. (2011).

The third stream of research that is relevant for our work studies inventory rationing and revenue management. Inventory rationing techniques support the allocation of inventory units among a heterogeneous customer base, by setting critical inventory levels and/or setting inventory control mechanisms. The concept of critical levels was introduced by Veinott (1965) and Topkis (1968) and since then, solutions have been provided for various control policies and demand classes. For a comprehensive literature review on inventory rationing we refer to Teunter and Klein Haneveld (2008). Although the concept of critical levels supports customer heterogeneity, it cannot easily be applied to our problem because of two reasons. First, critical level literature usually assumes a single source location. Sourcing flexibility–a key characteristic of our problem–is not accounted for. Second, the concept of critical levels offers only limited opportunities for customer differentiation in spare parts settings like ours where stock levels are low and the number of customer classes is usually more than two.

To overcome these difficulties, Jalil (2011) follows the concepts of revenue management to formulate a problem similar to ours as a multi-period MDP. The relative value function is approximated using linear programming. In the linear program, remaining stock at the end of the horizon has no value, future replenishments are ignored, and demand is assumed to be deterministic. In numerical experiments, he shows that in situations where customer heterogeneities are high and actual stock levels are low the revenue management heuristic achieves significant lower total cost over the next 25 time periods than the static allocation rule. What differentiates our work from Jalil (2011) is that we approximate the relative value function using an infinite time horizon taking into account future replenishments and honoring the stochastic nature of demand.

3. Problem description and model formulation

In this section we define the inventory control problem, discuss assumptions, introduce our notation, and formulate our model.

3.1. Problem description and notation

We consider a single item, single echelon, spare parts inventory network consisting of multiple local warehouses where spare parts are kept on stock for customers with different service contracts. Whenever a part at a customer site fails, it has to be replaced by a spare part. Part requests can be fulfilled from every warehouse in the network with stock at hand. The central warehouse keeps spare parts to replenish the local warehouses, and has also access to an emergency delivery mode to fulfill customer demand. The delivery time and the delivery cost from a warehouse to a customer are fixed and depend on the geographical coordinates and the delivery mode (emergency or regular). Typically, emergency deliveries are significantly more expensive than regular deliveries. Customers have service contracts with committed response and repair targets backed by penalties in case of contract violations. Penalty cost grows linearly in the delivery time beyond the service deadline, and depend on the contract type. Typically, the hourly penalty cost rate for 2 hours service contracts is (much) higher than for 8 hours contracts. Our objective is to minimize the sum of the average annual delivery cost and the average annual penalty cost.

3.2. Discussion of main assumptions

(i) The central warehouse has ample stock. The reason for this assumption is that replenishment decisions at the central warehouses are often decoupled from replenishment and allocation decisions at lower network echelons. In real life, we often see that the central warehouse can obtain new items from multiple channels such as regular suppliers, emergency suppliers, repair, and assembly/production facilities.

(ii) Part requests from each customer follow an independent Poisson process. The Poisson assumption is common in spare parts logistics.

(iii) Replenishment lead times are exponentially and identically distributed and have the same mean for all local warehouses. The assumption of equal replenishment lead times is justified because in real-life replenishment lead times are simply fixed at the same values for all local warehouses in the same geographical region. The assumption of an exponential shape of the replenishment lead time distribution is made to facilitate an exact MDP analysis for problem instances of sufficiently small size. Alfredsson and Verrijdt (1999) showed that the performance of a model with a static allocation rule in a complete pooling situations is rather insensitive to the choice of exponential or constant replenishment lead times. It is reasonable to assume that this also holds for our model. However, it is also clear that in situations with constant replenishment lead times, information on remaining replenishment lead times can be useful to calculate smart allocation decisions. We leave this for further research.

(iv) Inventories at local warehouses are controlled through continuous-time base stock policies with given base stock levels. In real life, stock levels are often reviewed only once a day. Since the replenishment lead time for a local warehouse is typically 3–6 days, we can however accurately approximate the periodic replenishment with a continuous replenishment with an adjusted average replenishment lead time. The assumption of base stock control at the local warehouses is justified because we consider inventory networks of expensive technical parts where holding cost is usually high and demand is usually low.

(v) Service deadlines, penalty cost, and delivery times are such that is never beneficial to backorder demand. Demand is thus always fulfilled immediately, either from a local warehouse or from the central warehouse (by means of an emergency delivery).

(vi) Customers in the same geographical area are aggregated into one customer region. Delivery times and delivery costs for all customers in the same customer region are the same. Demand rates for spare parts are available for each pair of customer region and contract type. This assumption makes sense because the expected life time of a spare part is large, and accurate information on the number and the condition of installed parts is usually not available. Hence, demand rate estimates are much more reliable on customer region level than on individual customer level.

The notation of our model is given in Table 1.

Because we assume one-for-one replenishment at all local warehouses, each spare part demand leads to one spare part leaving the central warehouse (either directly as an emergency delivery to the customer or as a replenishment shipment to the local warehouse that fulfills the customer demand). Consequently, replenishment cost at the central warehouse can be ignored because it does not depend on the allocation decision. Together with assumption (v) this implies that we can pre-calculate the total cost for fulfilling a part request from customer region j and customer class k from stocking location i according to:

Table 1
Notation.

Indices	
$i = 0,\ldots,I$	Stocking locations; $i = 0$ refers to the central warehouse
$j = 1,\ldots,J$	Customer regions
$k = 1,\ldots,K$	Customer classes (defined over contract types)
Parameters	
S_i	Base stock level of stocking location $i = 1,\ldots,I$
$\frac{1}{\mu}$	Average replenishment lead time of a local warehouse
λ_{jk}	Demand rate of customer region j and customer class k
W_k^{max}	Maximum response time for customer class k
t_{ij}	Delivery time from stocking location i to customer region j
C_{ij}^d	Delivery cost to ship one item from stocking location i to customer region j
C_k^p	Hourly penalty cost for contract violations of customer class k
C_i^r	Unit replenishment cost for stocking location $i > 0$
C_{ijk}^f	Total cost to fulfill a demand from customer region j and customer class k from stocking location i

$$C_{ijk}^f = \begin{cases} C_{ij}^d + C_k^p \cdot (t_{ij} - W_k^{max})^+ & \text{if } i = 0 \text{ (central warehouse)} \\ C_{ij}^d + C_k^p \cdot (t_{ij} - W_k^{max})^+ + C_i^r & \text{if } i > 0 \text{ (all local warehouses)} \end{cases}$$

3.3. MDP formulation

In this section we formulate the real-time allocation problem as a continuous-time average cost MDP with finite state and control spaces (see e.g. Bertsekas, 2007, pp. 310–316). State transitions and action selections take place at time instances when one of the following two event types occurs: (i) a replenishment order arrives at a local warehouse, or (ii) a customer issues a new part request. Times between successive transitions have an exponential probability distribution.

We describe the state of the system by $\mathbf{x} = (\mathbf{z},j,k)$, with $\mathbf{z} = (z_1,\ldots,z_I)$ the I-dimensional vector of actual stock levels at the local warehouses, and (j,k) references to the customer region and the customer class associated with the arriving part request. If the state refers to a replenishment order arrival, we set $j$and $k$equal to 0. We define the state space $\mathcal{S}$ as $\mathcal{S} = \{((z_1,\ldots,z_I),j,k)|z_i \in \{0,\ldots,S_i\}, i = 1,\ldots,I, j = 1,\ldots,J, k = 1,\ldots,K\} \cup \{((z_1,\ldots,z_I),0,0)|z_i \in \{0,\ldots,S_i\}, i = 1,\ldots,I\}$.

We now define the action space $\mathcal{A}$, and the set of admissible actions $\mathcal{A}(\mathbf{x})$ for each state $\mathbf{x} \in \mathcal{S}$. The action space for our model is $\mathcal{A} = \{-1,0,\ldots,I\}$. Here, action $i \geqslant 0$ stands for the decision to send a spare part from warehouse i to the customer who has just issued a part request, and action -1 stands for the decision to do nothing. Obviously, we have that $\mathcal{A}(\mathbf{z},0,0) = \{-1\}$ for all $\mathbf{z}$. The set of admissible actions for a state that represents a demand arrival, consists of all local warehouses with stock at hand plus the central warehouse: $\mathcal{A}(\mathbf{z},j,k) = \{i|z_i > 0, i = 1,\ldots,I\} \cup \{0\}$ for all $(\mathbf{z},j,k) \in \mathcal{S}$ with $j,k > 0$.

Next, we describe the transitions for our continuous-time MDP formulation. We assume that if the system is in state $\mathbf{x}$ and action a is applied, the next state will be $\mathbf{y}$ with probability $p_{\mathbf{x},\mathbf{y}}(a)$. The probabilities $p_{\mathbf{x},\mathbf{y}}(a)$ are called transition probabilities (see Bertsekas, 2007, p. 306). Furthermore, we define $\mathbf{e}_i$ for $i > 0$ as the I-dimensional unit vector with a 1 at position i and $\mathbf{e}_0$ as the zero vector, i.e. $\mathbf{e}_0 = \mathbf{0}$.

Transition type 1: initial state: $\mathbf{x} = (\mathbf{z},0,0)$, action: -1, next event: part request from customer region n and customer class p, next state: $\mathbf{y} = (\mathbf{z},n,p)$. The transition rate is λ_{np} and the transition probability $p_{\mathbf{x},\mathbf{y}}(-1)$ is equal to $\lambda_{np}\Big/\Big[\sum_{t=1}^{J}\sum_{u=1}^{K}\lambda_{tu} + \mu\sum_{r=1}^{I}(S_r - z_r)\Big]$.

Transition type 2: initial state: $\mathbf{x} = (\mathbf{z},0,0)$, action: -1, next event: order arrival at local warehouse m, next state: $\mathbf{y} = (\mathbf{z} + \mathbf{e}_m,0,0)$. The transition rate is $\mu(S_m - z_m)$ and the transition probability $p_{\mathbf{x},\mathbf{y}}(-1)$ is equal to $\mu(S_m - z_m)\Big/\Big[\sum_{t=1}^{J}\sum_{u=1}^{K}\lambda_{tu} + \mu\sum_{r=1}^{I}(S_r - z_r)\Big]$.

Transition type 3-a: initial state: $\mathbf{x} = (\mathbf{z},j,k)$ with $(j,k) \in \{1,\ldots,J\} \times \{1,\ldots,K\}$, action: 0, next event: part request from customer region n and customer class p, next state: $\mathbf{y} = (\mathbf{z},n,p)$. The transition rate is λ_{np} and the transition probability $p_{\mathbf{x},\mathbf{y}}(0)$ is equal to $\lambda_{np}\Big/\Big[\sum_{t=1}^{J}\sum_{u=1}^{K}\lambda_{tu} + \mu\sum_{r=1}^{I}(S_r - z_r)\Big]$.

Transition type 3-b: initial state: $\mathbf{x} = (\mathbf{z},j,k)$ with $(j,k) \in \{1,\ldots,J\} \times \{1,\ldots,K\}$, action: i with $i \in \{1,\ldots,I\}$ and $z_i > 0$, next event: part request from customer region n and customer class p, next state: $\mathbf{y} = (\mathbf{z} - \mathbf{e}_i,n,p)$. The transition rate is λ_{np} and the transition probability $p_{\mathbf{x},\mathbf{y}}(i)$ is equal to $\lambda_{np}\Big/\Big[\sum_{t=1}^{J}\sum_{u=1}^{K}\lambda_{tu} + \mu + \mu\sum_{r=1}^{I}(S_r - r_m)\Big]$.

Transition type 4-a: initial state: $\mathbf{x} = (\mathbf{z},j,k)$ with $(j,k) \in \{1,\ldots,J\} \times \{1,\ldots,K\}$, action: 0, next event: order arrival at local warehouse m, next state: $\mathbf{y} = (\mathbf{z} + \mathbf{e}_m,0,0)$. The transition rate is $\mu(S_m - z_m)$ and the transition probability $p_{\mathbf{x},\mathbf{y}}(0)$ is $\mu(S_m - z_m)\Big/\Big[\sum_{t=1}^{J}\sum_{u=1}^{K}\lambda_{tu} + \mu\sum_{r=1}^{I}(S_r - z_r)\Big]$.

Transition type 4-b: initial state: $\mathbf{x} = (\mathbf{z},j,k)$ with $(j,k) \in \{1,\ldots,J\} \times \{1,\ldots,K\}$, action: i with $i \in \{1,\ldots,I\}$ and $z_i > 0$, next event: order arrival at local warehouse m, next state: $\mathbf{y} = (\mathbf{z} - \mathbf{e}_i + \mathbf{e}_m,0,0)$. If $m \neq i$, the transition rate is $\mu(S_m - z_m)$ and the transition probability $p_{\mathbf{x},\mathbf{y}}(i)$ is $\mu(S_m - z_m)\Big/\Big[\sum_{t=1}^{J}\sum_{u=1}^{K}\lambda_{tu} + \mu + \mu\sum_{r=1}^{I}(S_r - z_r)\Big]$. If $m = i$, the transition rate is $\mu(S_i + 1 - z_i)$ and the transition probability is $\mu(S_i + 1 - z_i)\Big/\Big[\sum_{t=1}^{J}\sum_{u=1}^{K}\lambda_{tu} + \mu + \mu\sum_{r=1}^{I}(S_r - z_r)\Big]$.

The mean transition period lengths $\tau(\mathbf{x},a)$ for all state/action pairs directly follow from the transition rates. For state $(\mathbf{z},0,0)$ and action -1 the transition period length $\tau((\mathbf{z},0,0),-1)$ is $1\Big/\Big[\sum_{t=1}^{J}\sum_{u=1}^{K}\lambda_{tu} + \mu\sum_{r=1}^{I}(S_r - z_r)\Big]$. For state $(\mathbf{z},j,k)$ with $(j,k) \in \{1,\ldots,J\} \times \{1,\ldots,K\}$ and action 0 the transition period length $\tau((\mathbf{z},j,k),0)$ is equal to $\tau((\mathbf{z},0,0),-1)$. Finally, for state $(\mathbf{z},j,k)$ with $(j,k) \in \{1,\ldots,J\} \times \{1,\ldots,K\}$ and action i with $i \in \{1,\ldots,I\}$ the transition period length $\tau((\mathbf{z},j,k),i)$ is $1\Big/\Big[\sum_{t=1}^{J}\sum_{u=1}^{K}\lambda_{tu} + \mu + \mu\sum_{r=1}^{I}(S_r - z_r)\Big]$. We conclude our MDP formulation with the specification of the expected direct cost $G((\mathbf{z},j,k),a)$ when choosing action a in state $(\mathbf{z},j,k)$:

$$G((\mathbf{z},j,k),a) = \begin{cases} C_{ajk}^f & \text{if } a \geqslant 0 \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

To obtain the optimal allocation rule we transform the continuous-time MDP into a discrete-time MDP by applying a technique

called *uniformization* (see e.g. Bertsekas, 2007, pp. 288–295). The uniformization procedure consists of two steps: In the first step, we determine a new transition period length τ such that $\tau \leqslant \tau(\mathbf{x},a)$ for all $\mathbf{x} \in \mathcal{S}$, $a \in \mathcal{A}$. It is easy to see that $\tau = 1\Big/\left[\sum_{t=1}^{J}\sum_{u=1}^{K}\lambda_{tu} + \mu\sum_{r=1}^{I}S_r\right]$ is an appropriate choice. In the second step, we add two types of transitions: (i) from state $(\mathbf{z},0,0)$ and action -1 to $(\mathbf{z},0,0)$, and (ii) from state $(\mathbf{z},j,k)$ and action $i \in \mathcal{A}(\mathbf{z},j,k)$ to $(\mathbf{z}-\mathbf{e}_i,0,0)$. The transition rates for these fictitious transitions are chosen such that $\tau(\mathbf{x},a) = \tau$ for all $\mathbf{x} \in \mathcal{S}$, and $a \in \mathcal{A}(\mathbf{x})$. We obtain the desired discrete-time MDP by replacing the exponentially distributed transition period lengths with constant transition period lengths with the same mean.

The optimal average cost $\lambda^\star$ can be obtained by solving the Bellman optimality equations for the (discrete-time) average cost MDP:

$$h^\star(\mathbf{x}) = \min_{a\in\mathcal{A}(\mathbf{x})}\left[G(\mathbf{x},a) - \lambda^\star\tau + \sum_{\mathbf{y}\in\mathcal{S}}p_{\mathbf{x},\mathbf{y}}(a)h^\star(\mathbf{y})\right] \quad \forall \mathbf{x}\in\mathcal{S} \tag{2}$$

With the condition $\mathbf{c}^\mathbf{T}\mathbf{h}^\star = 0$ (for any $\mathbf{c}^\mathbf{T} \geqslant \mathbf{0}$), Eq. (2) has a unique finite solution and the optimal action $a^\star(\mathbf{x})$ is the action that attains the minimum in (2). $\mathbf{h}^\star$ is known as the optimal relative (or differential) cost vector and $h^\star(\mathbf{x}) - h^\star(\mathbf{y})$ represents the expected cost difference over an infinite time horizon under the optimal policy when starting in state $\mathbf{x}$ instead of $\mathbf{y}$. We solve (2) using relative value iteration (see e.g. Bertsekas, 2007, pp. 204–229). If we set $\mathbf{c}^\mathbf{T}$ equal to the state probabilities, the value of $h^\star(\mathbf{x})$ can be interpreted as the total additional cost over an infinite time horizon when starting in state $\mathbf{x}$ compared to paying the average cost $\lambda^\star$ every time unit. This interpretation of $h^\star(\mathbf{x})$ plays an important role in the development of our dynamic allocation rule in the next section.

4. Heuristic allocation rules

In this section, we present three heuristic allocation rules for the spare parts inventory network discussed in the previous section. The first heuristic allocation rule is a simple static allocation rule (denoted as SA-rule) which is described in Section 4.1. Because this rule is widely used in real life, it constitutes an important benchmark for any newly developed allocation rule. Next, we move to dynamic allocation rules. In this paper, we restrict ourselves to *one-step lookahead (1SL) policies*. One-step lookahead policies choose at each state $\mathbf{x}$ the action $a^{1SL}(\mathbf{x})$ that minimizes the sum of the direct cost and an approximation of the optimal future cost:

$$a^{1SL}(\mathbf{x}) = \arg\min_{a\in\mathcal{A}(\mathbf{x})}\left[G(\mathbf{x},a) + \sum_{\mathbf{y}\in\mathcal{S}}p_{\mathbf{x},\mathbf{y}}(a)\hat{h}(\mathbf{y})\right] \quad \forall \mathbf{x}\in\mathcal{S} \tag{3}$$

If $\hat{\mathbf{h}}$ is the relative cost vector of some heuristic allocation rule (called the *base policy*), then the one-step lookahead policy is called a *rollout policy*. For rollout policies, we can use the Bellman equations for the base policy (see Bertsekas, 2007, p. 198) to simplify (3) to:

$$\begin{aligned} a^{1SL}(\mathbf{z},j,k) &= \arg\min_{a\in\mathcal{A}(\mathbf{z},j,k)}\left[C^f_{ajk} + \sum_{\mathbf{y}\in\mathcal{S}}p_{(\mathbf{z},j,k),\mathbf{y}}(a)\hat{h}(\mathbf{y})\right] \\ &= \arg\min_{a\in\mathcal{A}(\mathbf{z},j,k)}\left[C^f_{ajk} + \sum_{\mathbf{y}\in\mathcal{S}}p_{(\mathbf{z}-\mathbf{e}_a,0,0),\mathbf{y}}(-1)\hat{h}(\mathbf{y})\right] \\ &= \arg\min_{a\in\mathcal{A}(\mathbf{z},j,k)}\left[C^f_{ajk} + \hat{h}(\mathbf{z}-\mathbf{e}_a,0,0) + \lambda^\star\tau\right] \\ &= \arg\min_{a\in\mathcal{A}(\mathbf{z},j,k)}\left[C^f_{ajk} + \hat{h}(\mathbf{z}-\mathbf{e}_a,0,0)\right] \quad \forall (\mathbf{z},j,k)\in\mathcal{S} \text{ and } j>0 \end{aligned} \tag{4}$$

The second heuristic allocation rule we present in this section is a rollout allocation policy with the SA-rule as base policy. This policy is denoted as the RA-rule and is described in Section 4.2. To derive the RA-rule, we must solve a system of $(J+1)\prod_{i=1}^{I}(S_i+1)$ Bellman equations in order to obtain $\mathbf{h}^{\mathbf{SA}}$, the relative cost vector under the SA-rule. For problem instances of real-life size this is infeasible. To overcome this problem, we develop a one-step lookahead allocation rule where we approximate the optimal relative cost with an estimate of the relative cost under the SA-rule. This third allocation rule is a dynamic allocation rule. It is denoted as the DA-rule and described in Section 4.3.

4.1. SA-rule

The SA-rule is a static allocation rule that uses predefined priorities to determine, among all warehouses with stock at hand, the warehouse that is selected to fulfill a part request. It takes as input $J \times K$ sorted lists of warehouses, one for each pair of customer region and customer class. Let $L_{jk}(q)$ denote the q-th warehouse in the static priority list of customer region j and customer class k. In case of a part failure at a customer of class k in customer region j, a new spare part is sent from the first warehouse in L_{jk} with stock at hand. The warehouses in L_{jk} are sorted in ascending order on the basis of the fulfillment cost C^f_{ijk}. This means that the SA-rule is a greedy allocation rule that selects among all warehouses with stock at hand the one with the lowest immediate fulfillment cost.

4.2. RA-rule

The RA-rule is a rollout policy that uses the SA-rule as base policy. An interesting property of rollout policies is that they have the cost improvement property which states that they achieve no worse results than their base policies. A major disadvantage is that the RA-rule requires calculating $\mathbf{h}^{\mathbf{SA}}$, and this involves solving a (possibly huge) set of Bellman equations. The practical value of the RA-rule is therefore limited. Yet, we have added the RA-rule to our analysis because it is the inspiration for our DA-rule.

4.3. DA-rule

In this subsection we present the DA-rule. The DA-rule is a one-step lookahead policy. In contrast to the optimal allocation rule and the RA-rule, it is applicable to problems of arbitrary size. First, we explain the general idea behind the DA-rule. Then, we discuss the three components that together make up the algorithm for approximating the optimal relative cost. We conclude with a formal description of the rule.

4.3.1. Idea behind the DA-rule

The DA-rule is a one-step lookahead policy where we approximate the optimal relative cost with an estimate of the relative cost under the SA-rule. The estimate is obtained through a computationally inexpensive iterative procedure. Consequently, the DA-rule can be considered as an approximation of the RA-rule (but one that can handle problems of arbitrary size).

Consider the system when a demand arrives. Without loss of generality, we assume that time is 0 and the actual stock level vector is equal to $\mathbf{z}$. We want to approximate $\mathbf{h}^{\mathbf{SA}}$ for all states $(\mathbf{z}-\mathbf{e}_a,0,0)$ with $z_a > 0$. Let us define $J(\mathbf{z}-\mathbf{e}_a,t_1,t_2)$ as the expected total cost during time interval $[t_1,t_2]$ if the system is in state $(\mathbf{z}-\mathbf{e}_a,0,0)$ at time 0 and the SA-rule is used to fulfill future demands. Then, by definition:

$$h^{SA}(\mathbf{z}-\mathbf{e}_a,0,0) = \lim_{t\to\infty}[J(\mathbf{z}-\mathbf{e}_a,0,t) - \lambda^{SA}t] \tag{5}$$

Our approximation of $\mathbf{h}^{SA}$ is based on a similar idea as in Axsäter (1990). We make a decomposition of the network into individual local warehouses. In this way, we only deal with one Markov process per local warehouse instead of one Markov process for the entire network. Under the SA-rule, all demand from customer region j and customer class k is first offered to warehouse $L_{jk}(1)$. If demand arrives when $L_{jk}(1)$ has no stock at hand, it is offered to $L_{jk}(2)$. If $L_{jk}(2)$ is also out of stock, it is offered to $L_{jk}(3)$, et cetera. We assume that the overflow demand stream from customer region j and customer class k to warehouse $i \neq L_{jk}(1)$ is a Poisson process. Consequently, each warehouse can be evaluated individually as an Erlang loss model ($M|M|S_i|S_i$ queue). In earlier contributions, Axsäter (1990), Alfredsson and Verrijdt (1999), Kukreja et al. (2001), Kutanoglu (2008), Kranenburg and Van Houtum (2009), and Reijnen et al. (2009) have derived approximations for the average network flow rates in steady state. However, since we want to calculate the relative cost, we are interested in transient system behavior. To the best of our knowledge this has not been looked at before.

We approximate the transient system behavior in the following way: We assume for each warehouse i a (constant) stockout probability p_i during $[0,T]$ and the steady state stockout probability $\bar{p}_i$ when $t > T$.

The parameter $T \geqslant 0$ is a design parameter for the DA-rule. In Section 4.3.2 we argue that setting T equal to the average replenishment lead time $\frac{1}{\mu}$ is a good choice for all problem instances. In the same subsection we also develop a procedure for calculating p_i. At this point, we just mention that the value of p_i depends (among other things) on the initial stock level vector $(\mathbf{z}-\mathbf{e}_a)$ and the value of T. In Section 5.2 Table 5 we verify our choice of T in a large numerical experiment. Next, we assume stationary demand streams during $[0,T]$ that follow from the stockout probabilities p_i, and stationary demand streams when $t > T$ that follow from the stockout probabilities $\bar{p}_i$. From the estimated stationary demand rates and the estimated stationary stockout probabilities we can immediately estimate the stationary flow rates in the network for $[0,T]$ and $t > T$. We can use these estimated stationary flow rates to derive approximations for $J(\mathbf{z}-\mathbf{e}_a,0,T)$ and $J(\mathbf{z}-\mathbf{e}_a,T,t)$ for arbitrary $t > T$. For the purpose of our analysis, we now rewrite (5) as:

$$h^{SA}(\mathbf{z}-\mathbf{e}_a,0,0) = J(\mathbf{z}-\mathbf{e}_a,0,T) + \lim_{t\to\infty}[J(\mathbf{z}-\mathbf{e}_a,T,t) - \lambda^{SA}t] \tag{6}$$

Let $\widehat{J}()$ denote our approximation of $J()$, and let $\hat{h}^{SA}(\mathbf{z}-\mathbf{e}_a)$ denote our approximation of $h^{SA}(\mathbf{z}-\mathbf{e}_a)$. Starting from Eq. (6) and using the assumption that the system is in steady state for $t > T$, we obtain $\hat{h}^{SA}(\mathbf{z}-\mathbf{e}_a,0,0) = \widehat{J}(\mathbf{z}-\mathbf{e}_a,0,T) - \lambda^{SA}T$. Because the relative cost vector $\mathbf{h}^{SA}$ is unique up to a constant (cf. the Bellman equations for a stationary policy; see Bertsekas (2007), p. 198), we may add $\lambda^{SA}T$ to all elements in $\hat{\mathbf{h}}^{SA}$ and simply define $\hat{h}^{SA}(\mathbf{z}-\mathbf{e}_a,0,0) = \widehat{J}(\mathbf{z}-\mathbf{e}_a,0,T)$. Plugging this into (4), we obtain the following expression for the DA-rule:

$$a^{DA}(\mathbf{z},j,k) = \arg\min_{a\in\mathcal{A}(\mathbf{z},j,k)}\left[C^f_{ajk} + \widehat{J}(\mathbf{z}-\mathbf{e}_a,0,T)\right]$$
$$\forall(\mathbf{z},j,k)\in\mathcal{S} \text{ and } j>0 \tag{7}$$

The algorithm for calculating $\widehat{J}(\mathbf{z}-\mathbf{e}_a,0,T)$ consists of three main steps. The first two steps are executed alternately and provide an accurate estimate of the average network flow during $[0,T]$ when starting at the actual stock level vector $(\mathbf{z}-\mathbf{e}_a)$ at time 0. Before we describe these two steps in detail, we introduce some notation: Let D_{ijk} denote (an approximation of) the average demand rate of customer class k in customer region j to warehouse i during $[0,T]$ and let p_i denote (an approximation of) the average stockout probability at warehouse i during $[0,T]$.

Step 1: Calculate the stockout probabilities p_i given the demand rates D_{ijk}, and the initial stock level vector $(\mathbf{z}-\mathbf{e}_a)$.
Step 2: Update the demand rates D_{ijk} given the average stockout probabilities p_i.
The two steps are executed until the changes in D_{ijk} in two consecutive iterations are smaller than some pre-specified, small value ϵ. The iterative process is initialized with $D_{ijk} = \lambda_{jk}$ if $i = L_{jk}(1)$ and $D_{ijk} = 0$ otherwise. In the third step, we calculate our approximation of the relative cost under the SA-rule for state $(\mathbf{z}-\mathbf{e}_a,0,0)$.
Step 3: Calculate $\widehat{J}(\mathbf{z}-\mathbf{e}_a,0,T)$, given the converged values of D_{ijk} and p_i.

We now explain these three steps in more detail.

4.3.2. Step 1: Calculating p_i for given demand rates D_{ijk}

Because we assume that the overflow demand streams are Poisson, we can model each warehouse i as an Erlang loss model with S_i servers, demand arrival rate $D_i = \sum_j\sum_k D_{ijk}$ and service rate μ. Suppose that the initial number of free servers (in our setting this is equivalent to the actual stock level) is equal to z_i. We are interested in approximating the average stockout probability during time interval $[0,T]$. The steady state stockout probability in an Erlang loss model can be calculated from the well-known Erlang B formula:

$$B\left(S_i,\frac{D_i}{\mu}\right) = \frac{\frac{1}{S_i!}\left(\frac{D_i}{\mu}\right)^{S_i}}{\sum_{x=0}^{S_i}\frac{1}{x!}\left(\frac{\lambda}{\mu}\right)^{x}} \tag{8}$$

Let $N(S_i,D_i,\mu,z_i,t)$ denote the expected number of rejected requests in the $M|M|S_i|S_i$ queue during time interval $[0,t]$ starting with z_i items in stock at time 0. Furthermore, let $\Delta(S_i,D_i,\mu,z_i)$ represent the additional number of rejected requests over an infinite time horizon starting with z_i items in stock at time 0, compared to steady state. The formal definition is given below:

$$\Delta(S_i,D_i,\mu,z_i) = \lim_{t\to\infty}\left[N(S_i,D_i,\mu,z_i,t) - D_i tB\left(S_i,\frac{D_i}{\mu}\right)\right] \tag{9}$$

Our idea is to approximate the transient stockout probability with a step-function. For $t \leqslant T$ we use a constant stockout probability p_i, and for $t > T$ we use the steady state stockout probability $\bar{p}_i = B\left(S_i,\frac{D_i}{\mu}\right)$. To calculate p_i we assume that the additional number of rejected requests compared to steady state all occur in the time interval $[0,T]$. Under this assumption the expected number of rejected requests during $[0,T]$ is equal to $D_iTB\left(S_i,\frac{D_i}{\mu}\right) + \Delta(S_i,D_i,\mu,z_i)$. Consequently, we can approximate the average stockouts probability during $[0\ T]$ as follows:

$$p_i = F(S_i,D_i,\mu,z_i,T)$$
$$= \max\left\{0,\min\left\{1,B\left(S_i,\frac{D_i}{\mu}\right) + \frac{\Delta(S_i,D_i,\mu,z_i)}{D_iT}\right\}\right\} \tag{10}$$

The min and max in expression (10) have been added to make sure that the calculated stockout probabilities are never smaller than zero or bigger than one. Before we can apply (10), we must calculate $\Delta(S_i,D_i,\mu,z_i)$.

To obtain $\Delta(S_i,D_i,\mu,z_i)$, the following steps are carried out: (i) we formulate the $M|M|S_i|S_i$ queue that represents warehouse i as an average cost MDP with a cost of 1 in case of a rejected customer demand, (ii) we construct a set of equations for the relative cost vector $\mathbf{h} = (h_0,h_1,\ldots,h_{S_i})$ consisting of (S_i+1) Bellman equations and one scaling equation, and (iii) we solve this system of equations. The scaling constraint is chosen such that $\Delta(S_i,D_i,\mu,z_i) = h_{z_i}$. We now explain these steps in more detail.

To formulate the $M|M|S_i|S_i$ queue as an average cost MDP we choose as state the number of items on stock. We have two events:

(i) arrival of a customer demand, and (ii) arrival of a replenishment order, and we also have two actions: (i) fulfill demand, and (ii) reject demand. For every state we have exactly one admissible action (defined by the nature of the $M|M|S_i|S_i$ queue): Demand is fulfilled if there is stock at hand, and rejected if there is no stock at hand. We incur unit cost for each rejected demand. The Bellman equations for this system read as:

$$h_0 = \frac{D_i}{D_i + S_i\mu} - \frac{D_i B\left(S_i, \frac{D_i}{\mu}\right)}{D_i + S_i\mu} + \frac{D_i}{D_i + S_i\mu} h_0 + \frac{S_i\mu}{D_i + S_i\mu} h_1$$

$$h_n = -\frac{D_i B\left(S_i, \frac{D_i}{\mu}\right)}{D_i + (S_i - n)\mu} + \frac{D_i}{D_i + (S_i - n)\mu} h_{n-1} + \frac{(S_i - n)\mu}{\lambda + (S_i - n)\mu} h_{n+1} \quad 1 \leqslant n \leqslant S_i - 1 \tag{11}$$

$$h_{S_i} = -\frac{D_i B\left(S_i, \frac{D_i}{\mu}\right)}{D_i} + h_{S_i - 1}$$

Next, we add a scaling equation to (11) such that the resulting system of equations has a unique solution with $\Delta(S_i, D_i, \mu, z_i) = h_{z_i}$. This constraint states that weighted sum of all vector elements of $\mathbf{h}$ must be equal to zero where the weights are equal to the state probabilities. It reads as:

$$\sum_{n=0}^{S_i} \left[\frac{\frac{1}{n!}\left(\frac{D_i}{\mu}\right)^n}{\sum_{x=0}^{S_i} \frac{1}{x!}\left(\frac{D_i}{\mu}\right)^x} \right] h_n = 0 \tag{12}$$

We obtain h_{z_i} (and thus $\Delta(S_i, D_i, \mu, z_i)$) by solving the system of equations consisting of (11) and (12). The simple structure of the system of equations allows for a fast sequential solution procedure. We have illustrated our step-function approximation of the real transient stockout probability for an arbitrary local warehouse in Example 1.

Example 1. Consider a local warehouse i with base stock level $S_i = 3$ and suppose that the average replenishment lead time $\frac{1}{\mu}$ is equal to $\frac{1}{5}$. Furthermore, suppose that at a certain point in the iterative procedure for calculating $\widehat{J}(\mathbf{z} - \mathbf{e}_a, 0, T)$, the demand arrival rate $D_i = \sum_j \sum_k D_{ijk}$ for warehouse i is equal to 12. Warehouse i can now be modeled as an $M|M|3|3$ queue with an average service time of $\frac{1}{5}$, and a demand arrival rate of 12. To investigate how the stockout probability at warehouse i evolves over time, we have written a Matlab simulation script. Fig. 1a shows the real stockout probability as a function of time for 0 items in stock at time 0 and Fig. 1b shows our approximation of the real stockout probability with $T = \frac{1}{\mu}$. Note that the shaded areas in Fig. 1a and b are equal.

Now, we motivate our choice to set T equal to $\frac{1}{\mu}$. Recall that T is used in the step-function approximation of the transient stockout probabilities, i.e. for each warehouse i, we assume a constant stockout probability p_i during $[0,T]$ and the steady state stockout probability $\bar{p}_i$ when $t > T$. In this subsection, we have developed a method for calculating p_i that uses the initial stock level vector $(\mathbf{z} - \mathbf{e}_a)$. It seems reasonable to choose T such that the impact of each candidate allocation decision a on the transient stockout probabilities is relatively strong during $[0,T]$ and relatively weak when $t > T$. Obviously, allocation decisions have most impact during the time period that they cause a (temporary) reduction of the stock level at the sourcing warehouse. Since we assume fixed base stock policies, replenishment orders are triggered immediately when a part leaves the warehouse. Consequently, it makes sense to set T equal to the average replenishment lead time $\frac{1}{\mu}$.

4.3.3. Step 2: Updating D_{ijk} for given average stockout probabilities p_i

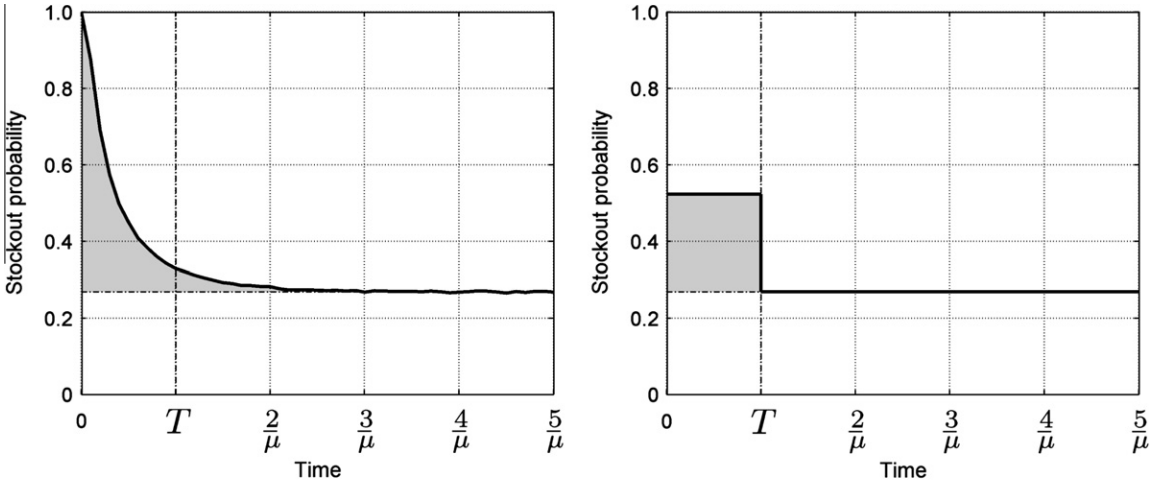
By definition of the SA-rule, all demand from customer region j and customer class k is offered to $L_{jk}(1)$. Suppose that warehouse $i = L_{jk}(1)$ is out of stock a fraction p_i of the time. Then, a fraction $(1 - p_i)$ of the demand from customer region j and customer class k is fulfilled from warehouse i, and an overflow demand stream $p_i \times \lambda_{jk}$ is offered to warehouse $L_{jk}(2)$. In general, the overflow demand rates D_{ijk} can be recursively determined from $D_{L_{jk}(q),j,k} = p_{L_{jk}(q-1)} D_{L_{jk}(q-1),j,k}$. Hence,

$$D_{L_{jk}(q),j,k} = \begin{cases} \lambda_{jk} & \text{if } q = 1 \\ \lambda_{jk} \prod_{n=1}^{q-1} p_{L_{jk}(q)} & \text{if } q > 1 \end{cases} \tag{13}$$

Note that in this recursive formula we use the simplifying assumption that actual stock levels at the local warehouses are independent.

4.3.4. Step 3: Calculating $\widehat{J}(\mathbf{z} - \mathbf{e}_a, 0, T)$ for given D_{ijk} and p_i

We can estimate the average flow rate of spare parts for customer class k during $[0,T]$ between warehouse i and customer region j by multiplying the (estimated) average demand rate during $[0,T]$ with the (estimated) average fill rate of warehouse i during $[0,T]$. We obtain an estimate of the average total cost during $[0,T]$ by taking the sum over all network edges of the product of the



(a) *Real transient behavior* (b) *Step-function approximation*

Fig. 1. Real and approximated transient stockout probability for $M|M|3|3$ queue and initial stock equal to 0.

Table 2
Parameter choices for the test beds.

Name of parameter	Values Experiment I	Values Experiment II
No. of local warehouses (I)	{6}	{24}
No. of customer regions (J)	{24}	{96}
Delivery time profile ($i > 0$)	$t_{ij} = 0.5 + 0.01 d_{ij}$	As Experiment I
Delivery cost profile ($i > 0$)	$C^d_{ij} = 1 d_{ij}$	As Experiment I
Emergency delivery cost (C^d_{0j})	{2000}	As Experiment I
Replenishment cost (C^r_i)	{0}	As Experiment I
Region indicator (R)	$\{r_1, r_2\}$	As Experiment I
$r_1 : \frac{1}{\mu} = 72, t_{0j} = 4$		
$r_2 : \frac{1}{\mu} = 120, t_{0j} = 8$		
Length service area unit (l)	$\left\{\sqrt{2}\cdot 150, 150, \frac{1}{3}\sqrt{6}\cdot 150\right\}$	As Experiment I
Customer class fractions ($\mathbf{w}$)	$\{(\frac{1}{6},\frac{2}{6},\frac{3}{6}),(\frac{2}{6},\frac{2}{6},\frac{2}{6}),(\frac{3}{6},\frac{2}{6},\frac{1}{6})\}$	As Experiment I
Relative network demand (ϕ)	{0.2, 0.5, 1.0}	As Experiment I
Penalty cost vector (C^p)	{(1200, 600, 300), (2400, 1200, 600), (4800, 2400, 1200)}	As Experiment I
Target time-based fill rate ($\boldsymbol{\Gamma}$)	X = {(0.5, 0.5, 0.5), (0.8, 0.8, 0.8), (0.95, 0.95, 0.95)}	$X \cup$ {(0.98, 0.98, 0.98)}

estimated flow rate (i.e. $(1 - p_i)D_{ijk}$), the fulfillment cost (i.e. C^f_{ijk}), and the length of the time horizon (i.e. T).

4.3.5. Formal specification

We conclude this subsection with a formal description of the DA-rule. It consists of the control rule (7) and the Algorithm 1 for calculating $\widehat{J}(\mathbf{z} - \mathbf{e}_a, 0, T)$. To evaluate the performance on problem instances of real-life size, we have implemented the DA-rule in a JAVA simulation program. Each time the simulator generates a new part request, we execute Algorithm 1 for all warehouses with stock at hand. No allocation decisions or relative cost vectors are stored. Consequently, storage space requirements are minimal. The average computation time per part request is less than 10 milliseconds on an Intel Pentium 4 2.16 gigahertz processor for all problem instances in our numerical experiments (including the ones of real-life size).

Algorithm 1. Calculate $\widehat{J}(\mathbf{y}, 0, T)$

initialization
$\forall (i,j,k) \in \{0,\ldots,I\} \times \{1,\ldots,J\} \times \{1,\ldots,K\}$ $D_{ijk} = 0$
$\forall (j,k) \in \{1,\ldots,J\} \times \{1,\ldots,K\}$ $D_{L_{jk}(1),j,k} = \lambda_{jk}$
$\forall i \in \{1,\ldots,I\}$ $D_i = \sum_{j=1}^{J}\sum_{k=1}^{K} D_{ijk}$
$p_0 = 0$
$\forall i \in \{1,\ldots,I\}$ $p_i = F(S_i, D_i, \mu, z_i, T)$ cf. (10)–(12)
repeat
$\forall (q,j,k) \in \{2,\ldots,I\} \times \{1,\ldots,J\} \times \{1,\ldots,K\}$ $D_{L_{jk}(q),j,k} = p_{L_{jk}(q-1)} \times D_{L_{jk}(q-1),j,k}$
$\forall i \in \{1,\ldots,I\}$ $D_i = \sum_{j=1}^{J}\sum_{k=1}^{K} D_{ijk}$
$\forall i \in \{1,\ldots,I\}$ $p_i = F(S_i, D_i, \mu, z_i, T)$ cf. (10)–(12)
until D_{ijk} does not change more than ϵ between two consecutive iterations
finalization
$\widehat{J}(\mathbf{y}, 0, T) = T\sum_{i=0}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} C^f_{ijk}(1 - p_i)D_{ijk}$

5. Numerical experiments

In this section, we investigate the performance and the structure of the proposed DA-rule via numerical experiments. We have the following objectives in conducting numerical experiments. First, we investigate the optimality gap of the DA-rule. Second, we investigate the impact of the two main approximation steps in the DA-rule: (i) using one-step lookahead with the SA-rule as base policy, and (ii) approximating the true relative cost under the SA-rule with $\widehat{J}(\mathbf{z} - \mathbf{e}_i, 0, T)$. Third, we verify that $\frac{1}{\mu}$ is an appropriate value for the design parameter T in the DA-rule. Our fourth objective is to characterize the DA-rule by investigating how it differs from the SA-rule. Finally, we investigate the potential cost savings of the DA-rule over the SA-rule on problem instances of real-life size, and we explore how the relative performance depends on various problem characteristics.

To meet the first four objectives we define a numerical experiment with a test bed containing a wide range of problem instances of small size (Experiment I). For all problem instances in this test bed, we can calculate the computationally expensive optimal allocation rule and the computationally expensive RA-rule. In this experiment we use relative value iteration to evaluate the performance of the allocation rules. To meet the fifth objective we define a numerical experiment with a test bed containing a wide range of problem instances of real-life size (Experiment II) and use discrete event simulation to evaluate the performance of the SA-rule and the DA-rule. In Section 5.1 we define the test beds, in Section 5.2 we describe the experiment with problem instances of small size, in Section 5.3 we characterize the DA-rule, and in Section 5.4 we describe the experiment with problem instances of real-life size.

5.1. Test bed

For Experiment I and Experiment II we use similar test beds based on full factorial designs on six parameters. Six more parameters are fixed within each test bed. The main difference between both experiments is that for Experiment I we create problem instances of small size (six local warehouses), whereas for Experiment II we create problem instances of real-life size (24 local warehouses). In both experiments, we consider three customer classes: customers with 2 hours contracts, customers with 4 hours contracts, and customers with 8 hours contracts. We start by summarizing our choices for all 12 parameters in Table 2. Costs are expressed in euros, times are expressed in hours, and distances are expressed in kilometers.

The total number of all possible combinations for these parameters is $2 \times 3 \times 3 \times 3 \times 3 \times 3 = 486$ for Experiment I, and $2 \times 3 \times 3 \times 3 \times 3 \times 4 = 648$ instances for Experiment II. We have randomly created 5 different sets of values for the coordinates of the customer regions, as there are uniform distributions involved in the generation of these values. This gives us in total $486 \times 5 = 2430$ instances for Experiment I and $648 \times 5 = 3240$ instances for Experiment II.

We now explain some of the parameters in more detail and describe how to construct problem instances from the parameters values. The first parameter in the factorial design is the region indicator (R). It can take two values r_1 or r_2. The value r_1 represents a situation where the service area is at relatively close distance to

the central warehouse; we use a small average replenishment lead time (72 hours) and a small emergency delivery time (4 hours). The value r_2 represents a situation where the service area is at relatively large distance from the central warehouse; we use a larger average replenishment lead time (120 hours) and a larger emergency delivery time (8 hours).

The second parameter in the factorial design is the length of a service area unit (l), which determines the network layout. For all problem instances, we create a rectangular service area consisting of 3×2 (Experiment I) or 6×4 (Experiment II) squares of $l \times l$ kilometers each. We position a local warehouse in the center of each square. For both experiments, the largest value of l is chosen such that an arbitrary part request at an arbitrary point inside the service area can in principle be fulfilled within the service deadline. Since we assume $t_{ij} = 0.5 + 0.01 d_{ij}$, we can travel 150 kilometers within 2 hours (=minimum service deadline). Consequently, the largest value for l in the factorial design is equal to $150\sqrt{2}$. For all problem instances we create $J = 4 \cdot I$ customer regions. Each customer region has coordinates that determine the delivery times and the delivery costs. Each customer region generates 2 hours, 4 hours, and 8 hours spare part demands. The location of each customer region is drawn according to a uniform distribution.

The third parameter in the factorial design is the weight vector $\mathbf{w} = (w_1, w_2, w_3)$ of the customer classes. It specifies for each customer region the fraction of demand associated with a service deadline of 2 hours, 4 hours, and 8 hours, respectively. In our experiments, we do not only assume that all customer regions have identical customer class weights, but also that their total demand rates are identical. A graphical illustration of a possible network layout for a problem instance in Experiment I with $l = 150\sqrt{2}$ is given in Fig. 2.

The fourth parameter in the factorial design is the average network demand during the replenishment lead time divided by the number of local warehouses. We refer to this parameter as the relative network demand (ϕ). From the factorial design parameters ϕ and $\mathbf{w}$ we obtain the problem inputs λ_{jk} according to:

$$\lambda_{jk} = \frac{w_k}{J} \phi I \mu$$

The fifth parameter in the factorial design is the hourly penalty cost vector C^p. For each customer class it specifies the penalty cost incurred per hour that a part request is fulfilled beyond the service deadline. Typically, the hourly penalty costs get bigger when the contractual maximum response times get smaller (so $C_1^p > C_2^p > C_3^p$).

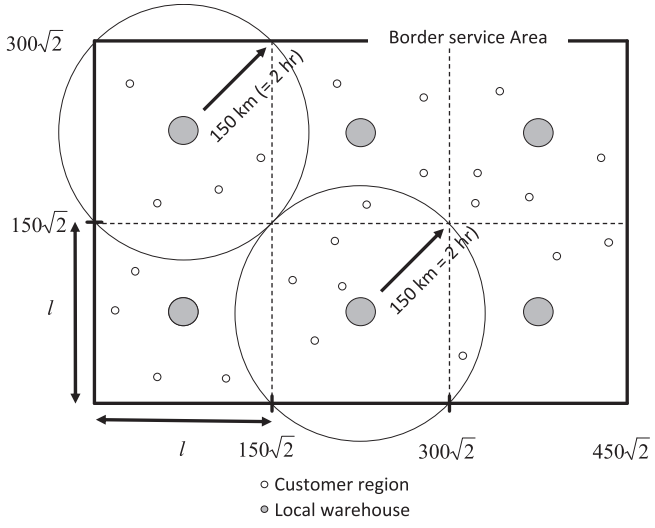


Fig. 2. Example of network layout in Experiment I.

The sixth and last parameter in the factorial design is the K-dimensional vector $\boldsymbol{\Gamma}$ which contains a target time-based fill rate for each customer class. We have developed a simple heuristic to calculate the base stock level vector $\mathbf{S}$ from the target fill rate vector $\boldsymbol{\Gamma}$. In this heuristic we make use of a method described in Reijnen et al. (2009) to approximate the average network flow rates when base stock levels are given and demand is fulfilled according to the SA-rule. We now summarize the heuristic (a formal description can be found in Appendix A).

In the first step, we set all base stock levels equal to zero. In the second step, we execute an iterative procedure where we increase the base stock level that minimizes the sum of the average annual delivery cost and the average annual penalty cost. We stop the iterative process when the time-based fill rates that follow from the method to approximate the average network flow rates are bigger than the target time-based fill rates specified in $\boldsymbol{\Gamma}$. So, by construction, the calculated base stock level vector $\mathbf{S}$ is close to optimal under the SA-rule and $|K|$ time-based fill rate constraints with bounds Γ_k, $k = 1, \ldots, K$. In real-life, OEMs typically set high target fill rates for SKU-s with low inventory holding costs, and (relatively) low target fill rates for SKU-s with high inventory holding costs. Thus via the target fill rates, we are able to create different problem instances that represent a wide spectrum of part types (expensive and inexpensive ones).

5.2. Experiment I

In this experiment we investigate the optimality gaps of the SA-rule, the RA-rule, and the DA-rule on the test bed defined in the previous subsection. For each of the 2430 problem instances, we have calculated the average annual total cost, the average annual regular delivery cost, the average annual emergency delivery cost, and the average annual penalty cost for all allocation rules by solving the balance equations for the Markov chain induced by the allocation rule. Consequently, all cost figures presented in this experiment are exact. We have run the numerical experiment on an Intel Pentium 4 2.16 gigahertz processor and 2.00 gigabyte RAM. Computation times for all allocation rules range from less than 0.1 second for problem instances with low base stock levels to 5–10 minutes for problem instances with high base stock levels ($1 \leqslant S_i \leqslant 5$).

We start by providing insight into the spread of the optimality gaps over the 2430 problem instances for the three heuristic allocation rules. For this purpose, we calculate for each allocation rule the 5%, 20%, 50%, 80%, and 95% percentile optimality gap, as well as the minimum and the maximum optimality gap. The results are shown in Table 3.

The table shows that the median optimality gap of the SA-rule is 4.4%. For 20% of all problem instances the optimality gap is more than 15%, and the maximum optimality gap is even more than 85%. So, in a considerable part of all problem instances there is big room for improvement. The numbers for the RA-rule look totally different. We see that for most problem instances the optimality gap of the RA-rule is less than one percent although the

Table 3
Experiment I: spread of optimality gap.

Percentile	Optimality gap (%)		
	SA-rule	RA-rule	DA-rule
Min	0.0	0.0	0.0
5%	0.3	0.0	0.0
20%	1.0	0.0	0.2
50%	4.4	0.0	0.9
80%	15.6	0.5	2.8
95%	37.0	4.3	5.9
Max	85.7	18.2	12.1

Table 4
Experiment I: optimality gap.

Test bed subset	N	Average optimality gap (%)		
		SA-rule	RA-rule	DA-rule
All instances	2430	**9.6** (−0.7, −5.7, 16.0)	**0.7** (−0.0, 2.3, −1.6)	**1.6** (0.3, −1.6, 3.0)
$R = r_1$	1215	**5.8** (−1.0, −3.9, 10.7)	**0.5** (0.1, 1.4, −1.0)	**1.2** (0.2, −1.5, 2.1)
$R = r_2$	1215	**13.3** (−0.5, −7.5, 21.3)	**0.9** (−0.1, 3.2, −2.2)	**2.1** (0.4, −1.8, 3.5)
$l = \sqrt{2} \cdot 150$	810	**8.0** (−1.7, −3.6, 13.3)	**0.5** (0.1, 1.5, −1.1)	**1.0** (0.2, −1.3, 2.1)
$l = 150$	810	**9.5** (−0.6, −5.8, 15.0)	**0.8** (−0.0, 2.5, −1.7)	**1.8** (0.3, −1.8, 3.3)
$l = \frac{1}{3}\sqrt{6} \cdot 150$	810	**11.2** (0.0, −7.6, 18.8)	**0.8** (−0.1, 2.9, −2.0)	**2.1** (0.3, −1.8, 3.6)
$\mathbf{w} = (\frac{1}{6}, \frac{2}{6}, \frac{3}{6})$	810	**7.4** (−0.1, −5.1, 12.6)	**0.8** (−0.0, 2.7, −1.6)	**1.4** (0.2, −1.2, 2.4)
$\mathbf{w} = (\frac{2}{6}, \frac{2}{6}, \frac{2}{6})$	810	**10.8** (−0.9, −6.3, 18.0)	**0.8** (−0.2, 2.7, −1.9)	**1.8** (0.3, −1.8, 3.3)
$\mathbf{w} = (\frac{3}{6}, \frac{2}{6}, \frac{1}{6})$	810	**10.6** (−1.2, −5.6, 17.4)	**0.5** (0.2, 1.5, −1.2)	**1.8** (0.4, −1.9, 3.3)
$\phi = 0.2$	810	**3.4** (−0.7, −2.8, 6.9)	**0.0** (0.0, 0.2, −0.2)	**0.5** (0.2, −0.3, 0.6)
$\phi = 0.5$	810	**9.1** (−0.9, −5.8, 15.8)	**0.5** (−0.0, 2.0, −1.5)	**1.6** (0.1, −1.7, 3.2)
$\phi = 1.0$	810	**16.1** (−0.6, −8.5, 25.3)	**1.6** (−0.1, 4.7, −3.1)	**2.8** (0.5, −2.9, 5.1)
$C^p = (1200, 600, 300)$	810	**2.8** (0.0, −2.6, 5.4)	**0.1** (0.0, 0.4, −0.4)	**0.9** (0.2, −1.3, 2.1)
$C^p = (2400, 1200, 600)$	810	**7.7** (−0.6, −5.3, 13.6)	**0.4** (−0.0, 1.9, −1.4)	**1.6** (0.2, −1.7, 3.1)
$C^p = (4800, 2400, 1200)$	810	**18.2** (−1.6, −9.2, 29.0)	**1.6** (−0.0, 4.7, −3.1)	**2.4** (0.5, −1.8, 3.7)
$\mathbf{\Gamma} = (0.50, 0.50, 0.50)$	810	**14.5** (0.6, −10.7, 24.6)	**1.2** (−0.4, 4.2, −2.6)	**1.4** (0.3, −1.7, 2.8)
$\mathbf{\Gamma} = (0.80, 0.80, 0.80)$	810	**9.7** (−1.4, −5.3, 16.4)	**0.7** (0.1, 2.4, −1.7)	**2.2** (0.3, −2.5, 4.4)
$\mathbf{\Gamma} = (0.95, 0.95, 0.95)$	810	**4.5** (−1.4, −1.1, 7.0)	**0.2** (0.3, 0.5, −0.5)	**1.3** (0.3, −0.7, 1.7)

maximum optimality gap of the RA-rule is considerable (18.2%). For the DA-rule, the 5%, 20%, 50%, 80%, and 95% optimality gap percentiles are a little bit higher than those of RA-rule, but still pretty small. A remarkable observation is that the maximum optimality gap of the DA-rule is smaller than the maximum optimality gap of the RA-rule. So apparently, the DA-rule can outperform the RA-rule for individual problem instances. Next, we calculate the average optimality gap for all three heuristic allocation rules over (i) all 2430 problem instances, and (ii) all subsets where one of the factorial design parameters takes one of its admissible values. Besides the average optimality gap (shown in bold face), we also show the contribution of the regular delivery cost, the emergency delivery cost, and the penalty cost to the optimality gap (these values are shown in parenthesis). The results are shown in Table 4.

The first row in Table 4 shows that the average optimality gap over all problem instances of the SA-rule, RA-rule, and DA-rule is 9.6%, 0.7%, and 1.6% respectively. The average optimality gap of the RA-rule is thus more than a factor 10 smaller than the average optimality gap of the SA-rule. This does not only hold for the average optimality gap over all problem instances, but also for the average optimality gap in 16 out of 17 subsets. From all this, we conclude that the RA-rule consistently performs very well on this test bed. Unfortunately, the RA-rule cannot be applied to problem instances of real-life size due to computational complexity (it requires solving a set of Bellman equations that grows exponentially in the number of local warehouses). That is why we have developed the DA-rule. The table also shows that the average optimality gap of the DA-rule is about a factor 2 bigger than the average optimality gap of the RA-rule. So apparently, approximating the true optimal relative cost $h^{\star}(\mathbf{z} - \mathbf{e}_i, 0, 0)$ with $h^{SA}(\mathbf{z} - \mathbf{e}_i, 0, 0)$, and approximating $h^{SA}(\mathbf{z} - \mathbf{e}_i, 0, 0)$ with $\widehat{J}\left(\mathbf{z} - \mathbf{e}_i, 0, \frac{1}{\mu}\right)$, both account for about half of the observed optimality gap of the DA-rule. When switching from the SA-rule to the DA-rule, we achieve an average cost reduction of $100\% \times (109.6 - 101.6)/109.6 = 7.3\%$. From the decomposition of the optimality gap into the three different cost components, we see that this cost reduction is mainly obtained by replacing regular deliveries by emergency deliveries (resulting in higher emergency delivery cost but lower penalty cost). In the next subsection, we investigate the differences between the SA-rule and the RA-rule in more detail. Finally, Table 4 also provides valuable information on the dependencies between the factorial design parameters and the optimality gaps for three heuristic allocation rules. The strongest dependencies are found for the relative network demand (ϕ), the penalty cost vector (C^p), and the target time-based fill rate vector ($\mathbf{\Gamma}$). For all three heuristic allocation rules, the dependencies point in the same direction, meaning that they all have relatively large optimality gaps for the same kind of problem instances. In particular, optimality gaps seem relatively large for problem instances where contract violations occur frequently (low $\mathbf{\Gamma}$) and are penalized strongly (big C^p), and the probability of multiple demands within one replenishment lead time is relatively high (big ϕ).

Table 5
Experiment I: impact of T on performance DA-rule.

	$T = \frac{1}{2} \cdot \frac{1}{\mu}$	$T = \frac{1}{\mu}$	$T = 2 \cdot \frac{1}{\mu}$	$T = 4 \cdot \frac{1}{\mu}$
Average optimality gap DA-rule (%)	2.4	1.6	2.8	4.1

To investigate the impact of the design parameter T on the performance of the DA-rule, we have compared the DA-rule with $T = \frac{1}{\mu}$ (our proposed value), with the DA-rules with $T = \frac{1}{2} \cdot \frac{1}{\mu}$, $T = 2 \cdot \frac{1}{\mu}$, and $T = 4 \cdot \frac{1}{\mu}$ on all 2430 problem instances of Experiment I. The DA-rule with $T = \frac{1}{\mu}$ does not only have the smallest average optimality gap over all 2430 problem instances (see Table 5), but also has the smallest average optimality gap in all 17 subsets defined by fixing one of the factorial design parameters to one of its admissible values. Our comparison showed that setting T equal to $\frac{1}{\mu}$ is a robust and appropriate choice for all problem instances. Therefore, we use $T = \frac{1}{\mu}$ in the remainder of this paper.

5.3. Characterization of DA-rule

In this subsection we aim to characterize the DA-rule. We do this by identifying the states where the DA-rule takes a different decision than the (simple) SA-rule. For our analysis we use the test bed of Experiment I. We start by decomposing the state space of each individual problem instance in disjoint sub spaces. The decomposition is obtained using four simple filters. A filter is like a decision node in classification trees. It takes as input a state space $\mathcal{S}$ and splits it in n disjoint sub state spaces $\mathcal{S}_1, \ldots, \mathcal{S}_n$ such that $\mathcal{S}_1 \cup \cdots \cup \mathcal{S}_n = \mathcal{S}$, based on the evaluation of the filter expression for all states $s \in \mathcal{S}$. The first filter evaluates for each state $(\mathbf{z}, j, k)$ the customer class k. The second filter evaluates for each state whether or not the DA-rule and the SA-rule propose the same deci-

Table 6
DA-rule characterization.

Subset no.	State space decomposition				Event probability	$q(S)$
	Filter 1	Filter 2	Filter 3	Filter 4		
1	$k = 1$	$a^{DA} = a^{SA}$	N/A	N/A	0.309	0
2	.	$a^{DA} \neq a^{SA}$	$a^{DA} = 0$	$z(a^{SA}) = 1$	0	414
3	.	.	.	$z(a^{SA}) > 1$	0	249
4	.	.	$z(a^{DA}) = 1$	$a^{SA} = 0$	0	N/A
5	.	.	.	$z(a^{SA}) = 1$	0.014	49
6	.	.	.	$z(a^{SA}) > 1$	0.001	18
7	.	.	$z(a^{DA}) > 1$	$a^{SA} = 0$	0	N/A
8	.	.	.	$z(a^{SA}) = 1$	0.007	68
9	.	.	.	$z(a^{SA}) > 1$	0.002	27
10	$k = 2$	$a^{DA} = a^{SA}$	N/A	N/A	0.276	0
11	.	$a^{DA} \neq a^{SA}$	$a^{DA} = 0$	$z(a^{SA}) = 1$	0.007	321
12	.	.	.	$z(a^{SA}) > 1$	0.000	202
13	.	.	$z(a^{DA}) = 1$	$a^{SA} = 0$	0	N/A
14	.	.	.	$z(a^{SA}) = 1$	0.031	55
15	.	.	.	$z(a^{SA}) > 1$	0.001	19
16	.	.	$z(a^{DA}) > 1$	$a^{SA} = 0$	0	N/A
17	.	.	.	$z(a^{SA}) = 1$	0.015	76
18	.	.	.	$z(a^{SA}) > 1$	0.003	28
19	$k = 3$	$a^{DA} = a^{SA}$	N/A	N/A	0.264	0
20	.	$a^{DA} \neq a^{SA}$	$a^{DA} = 0$	$z(a^{SA}) = 1$	0.027	850
21	.	.	.	$z(a^{SA}) > 1$	0.001	608
22	.	.	$z(a^{DA}) = 1$	$a^{SA} = 0$	0	N/A
23	.	.	.	$z(a^{SA}) = 1$	0.024	49
24	.	.	.	$z(a^{SA}) > 1$	0.001	18
25	.	.	$z(a^{DA}) > 1$	$a^{SA} = 0$	0	N/A
26	.	.	.	$z(a^{SA}) = 1$	0.014	72
27	.	.	.	$z(a^{SA}) > 1$	0.002	27

sion. For states where the two rules propose the same decision, we do not decompose further. For all other states, we apply a third and a fourth filter. The third filter considers for each state the associated DA decision and distinguishes between three options: (i) the DA-rule chooses the central warehouse, (ii) the DA-rule chooses a local warehouse with exactly one item on stock, and (iii) the DA-rule chooses a local warehouse with more than one item on stock. The fourth filter is similar to the third, but considers the SA decision instead of the DA decision. These four filters decompose the state space of each problem instance into $(3 \times 1 \times 1) + (3 \times 1 \times (9-1)) = 27$ disjoint sub spaces.

By solving the balance equations for the Markov chain induced by the DA-rule, we obtain the fraction of all decision events that belong to each of these 27 sub spaces. This provides valuable information on when and how often the DA-rule and the SA-rule propose different decisions. When the two rules propose different decisions, we investigate how the DA-rule rates the decision proposed by the SA-rule. For this purpose, we introduce the variable $q(x,y,\mathbf{z},j,k) = \left[C^f_{yjk} + h^{DA}(\mathbf{z} - \mathbf{e}_y, 0, 0)\right] - \left[C^f_{xjk} + h^{DA}(\mathbf{z} - \mathbf{e}_x, 0, 0)\right]$. We can interpret $q(x,y,\mathbf{z},j,k)$ as the cost difference (according to the DA-rule) when fulfilling a demand from customer region j and customer class k from warehouse x instead of warehouse y if the actual stock level vector is $\mathbf{z}$. We define the cost difference $q(\mathcal{S})$ over a set of states $\mathcal{S}$ as the weighted sum of all $q(a^{DA}(\mathbf{z},j,k),a^{SA}(\mathbf{z},j,k),\mathbf{z},j,k)$ with $(\mathbf{z},j,k) \in \mathcal{S}$. Let $\pi(\mathbf{z},j,k)$ denote the steady state probability of state $(\mathbf{z},j,k)$ under the DA-rule. Then, we get:

$$q(\mathcal{S}) = \frac{\sum_{(\mathbf{z},j,k)\in\mathcal{S}} \pi(\mathbf{z},j,k) q(a^{DA}(\mathbf{z},j,k), a^{SA}(\mathbf{z},j,k), \mathbf{z},j,k)}{\sum_{(\mathbf{z},j,k)\in\mathcal{S}} \pi(\mathbf{z},j,k)} \quad (14)$$

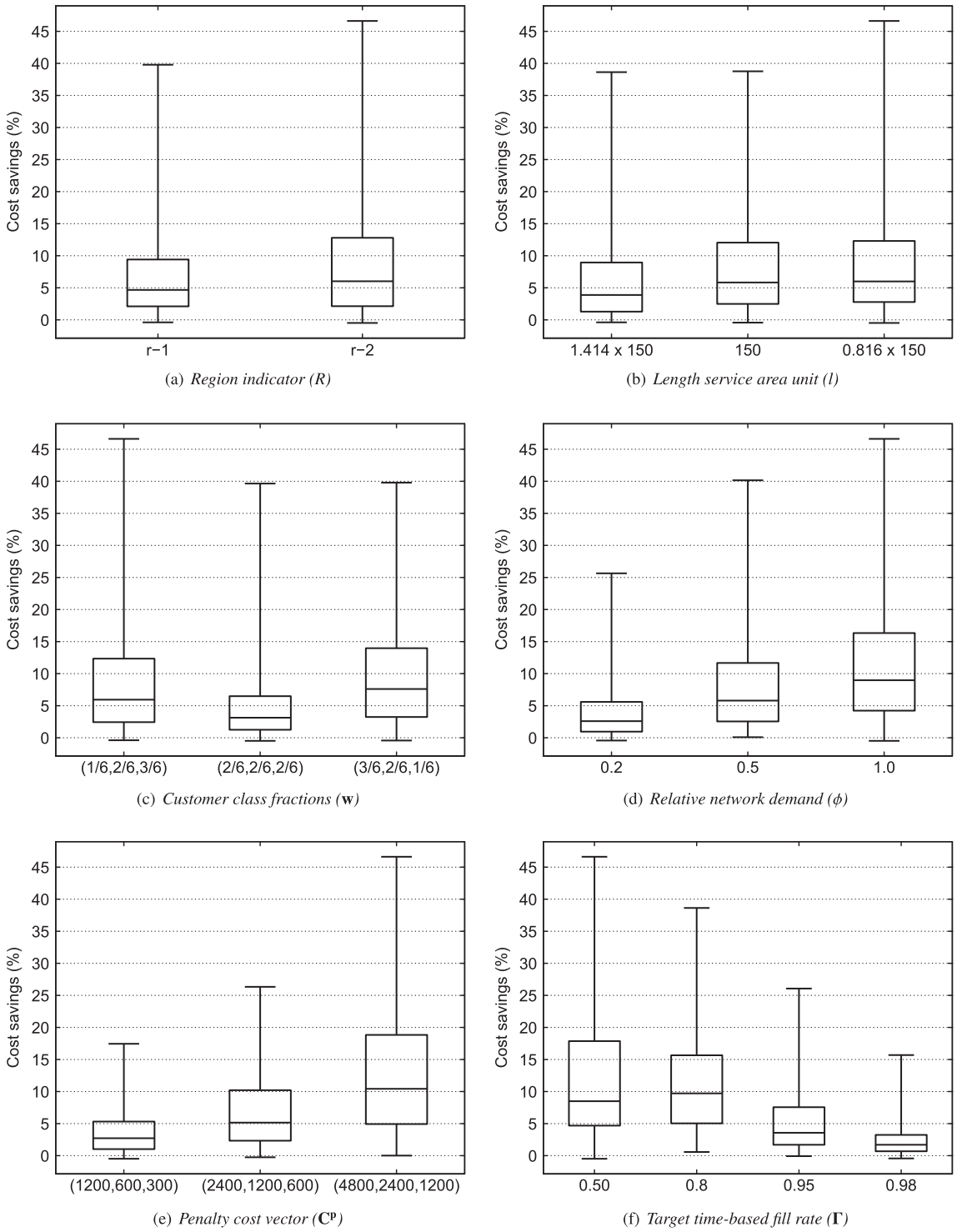
In Table 6 we have shown the average over all 2430 problem instances of the decision event probabilities and $q(\mathcal{S})$ for all 27 sub spaces. In line with previous notation, a^{DA} (a^{SA}) represents the warehouse selected by the DA (SA) rule, and $z(i)$ represents the actual stock level at warehouse i. From Table 6 we learn that on average for $100\% \times (0.309 + 0.276 + 0.264) = 84.9\%$ of all part requests the DA-rule and the SA-rule propose the same allocation decision. From subset 2 and 3 we learn that the DA-rule will only execute an emergency delivery for 2 hours demand if there is no other option.

Subsets 11 and 20 indicate that for 4 and 8 hours demand this is different; here the DA-rule prefers an emergency delivery from the central warehouse over a regular delivery from a local warehouse with only one item on stock. The explanation is that the DA-rule anticipates situations where a 2 hours demand occurs before the triggered replenishment order arrives and finds all nearby local warehouses out of stock.

The figures for subsets 8, 17, and 26 show that it also happens that the DA-rule decides to fulfill demand from a local warehouse with more than one item on stock whereas the SA-rule decides to select a local warehouse with only one item on stock. Furthermore, we see that the DA-rule expects the biggest cost reductions in situations where the DA-rule decides to fulfill an 8 hours demand from the central warehouse and the SA-rule decides to fulfill this demand from a local warehouse with only one item on stock (cf. subset 20; it has the highest $q(\mathcal{S})$ value of all subsets). Because the average event probability for subset 20 is also relatively high (2.7%), the decisions covered by this subset seem to be responsible for a large part of the cost reductions that can be achieved when switching from the SA-rule to the DA-rule. Summarizing we can say that the main difference between the DA-rule and the SA-rule is that the DA-rule is more reluctant to take away the last item at a local warehouse and less reluctant to use emergency deliveries from the central warehouse. A major strength of the DA-rule is that the decision whether or not to fulfill (an 8 hours) demand from a local warehouse with only one item on stock may depend on the on hand stock levels at neighboring local warehouses.

5.4. Experiment II

In this experiment we compare the DA-rule and the SA-rule on a test bed containing problem instances of real-life size. The test bed



(a) *Region indicator (R)*

(b) *Length service area unit (l)*

(c) *Customer class fractions* ($\mathbf{w}$)

(d) *Relative network demand* (ϕ)

(e) *Penalty cost vector* ($\mathbf{C}^{\mathbf{p}}$)

(f) *Target time-based fill rate* ($\mathbf{\Gamma}$)

Fig. 3. Experiment II: Box diagrams for parameters in factorial design.

has been defined in Section 5.1. For each of the 3240 problem instances, we have calculated the average annual total cost, the average annual regular delivery cost, the average annual emergency delivery cost, and the average annual penalty cost for the DA-rule, and the SA-rule. In contrast to Experiment I, we cannot evaluate the performance of the DA-rule and the SA-rule analytically and thus we move to discrete event simulation. See Section 5.4.1 for the details of how the simulation has been set up.

The average cost reduction over all 3240 problem instances when switching from the SA-rule to the DA-rule is 7.9%. When leaving out all problem instances with $\mathbf{\Gamma} = (0.98, 098, 0.98)$ (this parameter value is not present in Experiment I), the average cost reduction is even 9.7%. This is more than 30% bigger than the average cost reduction in Experiment I (7.3%) and a clear indication that the DA-rule scales well. To investigate how the cost savings depend on the various problem characteristics, we have created a

box diagram for each parameter in the factorial design. The box diagrams are shown in Fig. 3a–f and show the median, the 25% and 75% percentiles, and the minimum and maximum cost reductions.

Fig. 3a–f gives a similar picture as the results in Experiment I (cf. Table 4). We see that the cost reduction for problem instances with region indicator $R = r_2$ (emergency delivery time of 8 hours) is bigger than the cost reduction for problem instances with region indicator $R = r_1$ (emergency delivery time of 4 hours). We also see that the cost reduction gets bigger if the service area gets smaller. Fig. 3c shows no clear correlation between the cost reduction and the weight vector $\mathbf{w}$ of the customer classes. Fig. 3d–f however, shows a strong correlation between the cost savings and the relative network demand ϕ, the penalty cost vector C^p, and the target time-based fill rate vector $\boldsymbol{\Gamma}$, just as in Experiment I.

All together, switching from the SA-rule to the DA-rule seems in particular beneficial in situations where: (i) contract violations are expensive and occur frequently, (ii) emergency deliveries involve high delivery and/or penalty cost, and (iii) the probability of multiple demands within one replenishment lead time is relatively high. In all these situations, the SA-rule suffers from the weakness that it does not anticipate future stockouts and the contract violations and penalty costs this may cause.

5.4.1. Discrete event simulation

Here, we describe how we have set up our discrete event simulations. Each simulation run is divided in 21 sub-runs of 5000 part failure events each. The first sub-run is used as a warming-up period. For all other 20 sub-runs we calculate the average annual total cost. From these 20 samples we calculate the sample mean and the coefficient of variation. If the coefficient of variation is smaller than 0.01/2 the simulation is terminated. Otherwise, we double the number of part failure events we want to simulate and merge the current 20 sub-runs into 10 new ones by merging sub-runs 1 with 2, 3 with 4, ..., and 19 with 20. Under some mild conditions this methods guarantees that the simulated average annual total cost does not differ more than 1% from its expected value with a probability of more than 95%. This method is called the method of non-overlapping batch means (NOBM). For a comprehensive description of NOBM we refer the reader to Steiger and Wilson (2001).

6. Conclusions

We conclude by summarizing our main results. We developed a dynamic rule for allocating available network stock to real-time part requests in a single echelon multi location spare parts network with multiple customer classes where part requests can often be fulfilled from more than one warehouse within the service deadline. Our dynamic allocation rule is a one-step lookahead policy that approximates the optimal relative cost with an estimate of the relative cost under a static allocation rule.

First, we showed on a test bed with small problem instances that the optimality gap of our dynamic allocation rule is small (1.6% on average). This is much smaller than the optimality gap of a simple but widely used static allocation rule (9.6% on average). We also showed that on problem instances of real-life size, the dynamic allocation rule achieves average cost savings of 7.9% in comparison to the static allocation rule. This indicates that the dynamic allocation rule scales very well. Second, we showed that the dynamic allocation rule in particular outperforms the static allocation rule when contract violations are expensive and occur frequently, emergency deliveries involve high delivery and/or penalty cost, and the probability of multiple demands within one replenishment lead time is relatively high. Third, we characterized our dynamic allocation rule and showed that it mainly differs from the static allocation rule in situations where the static allocation rule selects a warehouse with only one item on stock, and the actual part request has a relatively high maximum response time. In particular, we showed that switching from the greedy static allocation rule to the dynamic allocation rule with (limited) lookahead changes the role of emergency deliveries from a tool of last resort to a tool that is actively used to avoid stock level reductions at local warehouses in situations where this is considered undesirable.

For future research, it is relevant to investigate the impact of demand forecast errors on the performance of the DA-rule. Another interesting direction for future research is the optimization of base stock levels under a dynamic allocation rule like the one presented in this paper. We plan to investigate this direction in the near future. We strongly believe that we can calculate excellent base stock levels using simulation-based optimization. In simulation-based optimization, simulation is used to evaluate the performance of base stock level vectors, and a standard optimization technique (e.g. local search) is used to find a good base stock level vector. When simulating the system to evaluate a base stock level vector, we allocate demand according to the DA-rule. Even for (very) large problem instances, simulation-based optimization seems feasible because: (i) the DA-rule does not contain any free parameters, (ii) the DA-rule performs very well for *arbitrary* base stock level vectors, and (iii) during each simulation, we can calculate on hand stock histograms for each local warehouse and use this information to steer the base stock level optimization.

Acknowledgments

The authors thank the anonymous referees for their helpful and valuable comments.

Appendix A. Calculation of base stock levels

In this appendix, we formally state the method for calculating a base stock level vector $\mathbf{S}$ from a target time-based fill rates vector $\boldsymbol{\Gamma}$. It consists of two parts. Algorithm 2 approximates the average cost per time unit and the time-based fill rates for given base stock level vectors. In Algorithm 3 we use this evaluation procedure to find a base stock level vector such that the (approximated) time-based fill rates exceed the target time-based fill rates, and the (approximated) average annual total cost is low.

Algorithm 2. Calculate $\widehat{C}(\mathbf{S})$ and $\widehat{\boldsymbol{\Gamma}}(\mathbf{S})$

initialization	
$\forall (i,j,k) \in \{0,\ldots,I\} \times \{1,\ldots,J\} \times \{1,\ldots,K\}$	$D_{ijk} = 0$
$\forall (j,k) \in \{1,\ldots,J\} \times \{1,\ldots,K\}$	$D_{L_{jk}(1),j,k} = \lambda_{jk}$
$\forall i \in \{1,\ldots,I\}$	$D_i = \sum_{j=1}^{J}\sum_{k=1}^{K} D_{ijk}$
	$p_0 = 0$
$\forall i \in \{1,\ldots,I\}$	$p_i = B\left(S_i, \frac{D_i}{\mu}\right)$ cf. (8)
repeat	
$\forall (q,j,k) \in \{2,\ldots,I\} \times \{1,\ldots,J\} \times \{2,\ldots,K\}$	$D_{L_{jk}(q),j,k} = p_{L_{jk}(q-1)} D_{L_{jk}(q-1),j,k}$
$\forall i \in \{1,\ldots,I\}$	$D_i = \sum_{j=1}^{J}\sum_{k=1}^{K} D_{ijk}$
$\forall i \in \{1,\ldots,I\}$	$p_i = B\left(S_i, \frac{D_i}{\mu}\right)$ cf. (8)
until D_{ijk} does not change more than ϵ between two consecutive iterations	
finalization	
$\widehat{C}(\mathbf{S}) = \sum_{i=0}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} C^f_{ijk}(1-p_i)D_{ijk}$	
$\widehat{\Gamma}_k(\mathbf{S}) = \left[\sum_{i=0}^{I}\sum_{j=1}^{J}(1-p_i)D_{ijk}\delta_{ijk}\right] \Big/ \left[\sum_{j=1}^{J}\lambda_{jk}\right]$	with $\delta_{ijk} = 1$ if $t_{ij} < W^k_{max}$ and 0 otherwise

Algorithm 3. Calculate $\mathbf{S}(\boldsymbol{\Gamma})$

initialization
$\forall i \in \{1,\ldots,I\}$ $S_i = 0$
calculate $\widehat{C}(\mathbf{S})$ and $\widehat{\boldsymbol{\Gamma}}(\mathbf{S})$ using Algorithm 2
$\forall i \in \{1,\ldots,I\}$ calculate $\Delta C(\mathbf{S},i) = C(\mathbf{S}) - C(\mathbf{S}+\mathbf{e_i})$
while $\exists k | \widehat{\Gamma}_k < \Gamma_k$ **do**
$i^\star = \text{argmax}_{i\in\{1,\ldots,I\}}[\Delta C(\mathbf{S},i)]$
$S_{i^\star} := S_{i^\star} + 1$
calculate $\widehat{C}(\mathbf{S})$ and $\widehat{\boldsymbol{\Gamma}}(\mathbf{S})$ using Algorithm 2
$\forall i \in \{1,\ldots,I\}$ calculate $\Delta C(\mathbf{S},i) = C(\mathbf{S}) - C(\mathbf{S}+\mathbf{e_i})$
end while

References

Alfredsson, P., Verrijdt, J., 1999. Modeling emergency supply flexibility in a two-echelon inventory system. Management Science 45, 1416–1431.

Axsäter, S., 1990. Modelling emergency lateral transshipments in inventory systems. Management Science 36, 1329–1338.

Axsäter, S., 2003. A new decision rule for lateral transshipments in inventory systems. Management Science 49, 1168–1179.

Bertsekas, D., 2007, . Dynamic Programming and Optimal Control, third ed., vol. II. Athena Scientific.

Jalil, M., 2011. Customer Information Driven After Sales Service Management: Lessons from Spare Parts Logistics. ERIM PhD Series Research in Management. Erasmus University Rotterdam.

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

Kukreja, A., Schmidt, C., Miller, D., 2001. Stocking decisions for low-usage items in a multilocation inventory system. Management Science 47, 1371–1383.

Kutanoglu, E., 2008. Insights into inventory sharing in service parts logistics systems with time-based service levels. Computers & Industrial Engineering 54, 341–358.

Minner, S., Silver, E., Robb, D., 2003. An improved heuristic for deciding on emergency transshipments. European Journal of Operational Research 148, 384–400.

Paterson, C., Kiesmuller, G., Teunter, R., Glazebrook, K., 2011. Inventory models with lateral transshipments: a review. European Journal of Operational Research 210, 125–136.

Reijnen, I., Tan, T., Van Houtum, G., 2009. Inventory planning for spare parts networks with delivery time requirements. Beta working paper 280. Eindhoven University of Technology, The Netherlands.

Steiger, N., Wilson, J., 2001. Convergence properties of the batch means method for simulation output analysis. INFORMS Journal on Computing 13, 277–293.

Teunter, R., Klein Haneveld, W., 2008. Dynamic inventory rationing strategies for inventory systems with two demand classes, poisson demand and backordering. European Journal of Operational Research 190, 156–178.

Topkis, D., 1968. Optimal ordering and rationing policies in a nonstationary dynamic inventory model with n demand classes. Management Science 15, 160–176.

Van Wijk, A., Adan, I., van Houtum, G., 2009. Optimal lateral transshipment policy for a two location inventory problem. Eurandom report 2009-027. Eindhoven University of Technology, The Netherlands.

Veinott Jr., A., 1965. Optimal policy for a multi-product, dynamic, nonstationary inventory problem. Management Science 12, 206–222.

Wong, H., van Houtum, G., Cattrysse, D., Van Oudheusden, D., 2005. Simple, efficient heuristics for multi-item multi-location spare parts systems with lateral transshipments and waiting time constraints. Journal of the Operational Research Society 56, 1419–1430.