International Journal of Production Research
Vol. 00, No. 00, 00 Month 0000, 1–22

Multi-Level Reverse Logistics Network Design Under Uncertainty

K. Lieckens[ab*] and N. Vandaele[ab⋆]

[a]*KU-Leuven, Faculty of Business Economics, Research Centre for Operations Management, Naamsestraat 69-3000 Leuven, Belgium*; [b]*KU-Leuven Campus Kortrijk, Faculty of Applied Economics, Etienne Sabbelaan 53-8500 Kortrijk, Belgium*

(*v3.0 released September 2010*)

Localising facilities and assigning product flows in a reverse logistics environment is a crucial but difficult strategic management decision, certainly when value decay plays an important part. Despite numerous publications regarding closed-loop supply chain design, very few addressed the impact of lead times and the high level of uncertainty in reverse processes. In this paper, a single product reverse logistics network design problem with multiple layers and multiple routings is considered. To this end, a new advanced strategic planning model with integrated queueing relationships is built that explicitly takes into account stochastic delays due to various processes like collection, production and transportation, as well as disturbances due to various sources of variability like uncertain supply, uncertain process times, unknown quality, breakdowns, etc. Their impact is measured by transforming these delays into work-in-process, which affects profit through inventory costs. This innovative modeling approach is difficult to solve because of both combinatorial and nonlinear continuous relationships. The differential evolution algorithm with an enhanced constraint handling method is proposed as an appropriate heuristic to solve this model close to optimality. A number of scenarios for a realistic case illustrate the power of this optimization tool.

Keywords: Differential evolution; Reverse flows; Supply chain network design; Queueing

*Corresponding author. Phone: +32 16 32 68 99. Email: kris.lieckens@econ.kuleuven.be; kris.lieckens@kuleuven-kortrijk.be

*Phone: +32 16 32 67 51. Email: nico.vandaele@econ.kuleuven.be; nico.vandaele@kuleuven-kortrijk.be

ISSN: XXXX-XXXX print/ISSN XXXX-XXXX online

DOI: XX.XXXX/XXXXXXXXXxxxxxxx
http://www.informaworld.com

1. Introduction

Given current economical and environmental concerns, it is of great importance to develop efficient Closed-Loop Supply Chains (CLSC). A CLSC integrates a reverse flow into the traditional forward chain in order to take back, re-process and re-sell product returns, either in the original primary market or in another secondary market. The question is where to locate all the required processes and how much product flow to assign to each process, such that a maximum profit and an excellent service in terms of delivery time can be guaranteed. As outlined in the extensive review of Rubio *et al.* (2008), this is a main topic in reverse logistics.

The research in this paper builds upon previous work of Lieckens and Vandaele (2007). The main contribution is the extension of the CLSC model towards network design decisions with multiple levels, quality dependent routings and stochastic transportation delays, while taking into account the interrelated queueing and variability effects. These extensions are useful because they all lead to a more realistic lead time. We develop an Advanced Strategic Planning model (ASP) that integrates financial information and relationships from queueing theory to decide on the appropriate location of multiple facilities in a CLSC, as well as the appropriate flow allocation of a single product type. A network design is considered as appropriate if it performs well with respect to revenue, costs and delays (i.e. delivery time). Being able to determine the optimal degree of flexibility is key for successful reverse logistics, especially in case of high recoverable value and a high decay rate. To this end, we transform expected, but variability dependent delays into expected inventory costs by using Little's law. This affects the net profit function, and thus the final network design decision. The price for better decisions is the introduction of highly nonlinear relationships in a mixed integer objective function, which complicates the search for optimal solutions. The second purpose of this research is to demonstrate that the Differential Evolution (DE) algorithm presented in Lieckens and Vandaele (2007) is still capable of finding good design solutions in case of multiple network levels, multiple routings and transportation issues. We propose an adapted algorithm scheme that handles the constraints differently and that decides on fractions to assign the flows to facilities instead of absolute values for the flows that enter and exit facilities. Using an examplary case study, we show that this procedure solves larger problems more efficiently. The structure and the size of the example are based on a case from practical industry.

The queueing analysis in our ASP Model is valid when the network is in a steady state condition. It implies that we only get an idea about an expected inventory level, i.e. a level that naturally arises at each location when the period under consideration is long enough. This prevents dynamic inventory control decisions like locating appropriate intermediate inventory buffers under a push versus a pull strategy, or specifying a maximum stock level with a reorder point, or coordinating returns with other supply sources such as new products. We refer to Fleischmann *et al.* (2003) for an inventory model where these features are integrated. They use a single-item base-stock model with backorders and a reorder-point order-up-to policy to calculate long-run average costs as a function of the target base-stock level.

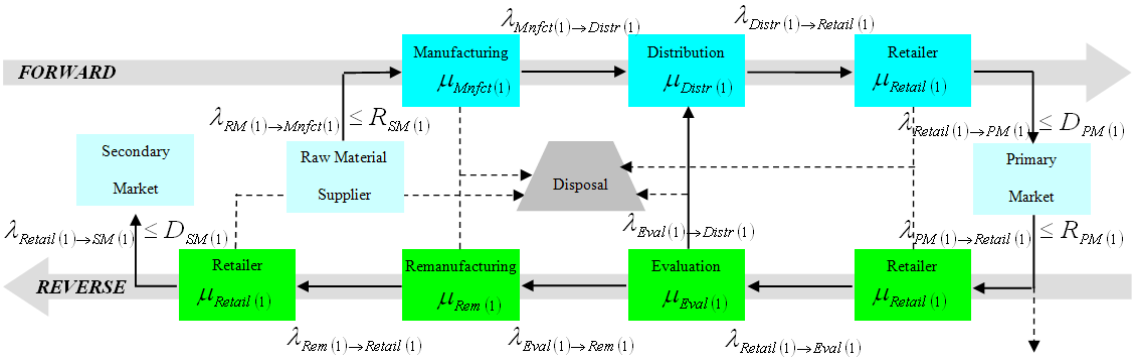

Figure 1. Structure of the CLSC.

2. Problem Description

In Section 3, we derive a generic ASP model that can be applied to any CLSC-structure. However, we will explain the network problem below and apply the ASP model in Section 5 using the structure of Guide *et al.* (2006), who developed a simple queueing network model that includes the marginal value of time to identify the drivers of reverse supply chain design. Their analysis of two companies in different industries, Hewlett-Packard in the inkjet printer industry and Bosch in the power tools industry, shows the impact of differences in processing and delay costs on the choice between an efficient and a responsive network. Key drivers of a highly responsive reverse supply chain are high return rates, high time value decay rates, low as-new return fractions, and substantial recoverable value.

The CLSC in Figure 1, where bold arrows represent the product flow and dashed arrows represent the disposal flow, includes forward chain nodes for manufacturing ($Mnfct$), distribution ($Distr$) and retailers in the primary market (PM), and reverse chain nodes for evaluation ($Eval$), remanufacturing (Rem) and retailers who collect and/or sell in the secondary market (SM). The retailer ($Retail$) is the single starting and ending node of the system. Disposal ($Disp$) is an option at each network level. Products are returned to the retailer where they are stored until a truck that is dedicated to a transportation line towards one of the evaluation centres can be loaded. All testing, evaluating and sorting activities are performed at this place. When a product return is labelled as a low grade, it is shipped to a remanufacturing facility, after which it is sold in the secondary market at a discount. A product return that is labelled as a high grade, only requires some minor reprocessing activities like repackaging, and is re-sold as-new in the primary market by using the forward distribution centre. Since a consumer is not able to distinguish the difference in quality, its selling price is the same as for the original new products, which are processed at the manufacturing facility. This distinction between high and low grades results in different routings that depend on the assessed quality. Examples include a printer classified as a low grade when the internal counter reveals that the number of printed pages exceeds some predefined treshhold (Guide *et al.* 2006), or dismantling electronic devices into spare parts for the open market and recyclable material fractions for external recyclers (Fleischmann *et al.* 2003), or a compressor classified as a high grade when the deviation between the two rotors is still within a specific target.

3. Model Formulation

3.1. *Product Arrivals*

Individual products return from the aggregate primary market, a single node $PM(1)$, towards a retailer node $Retail(j)$ according to a rate $\lambda_{PM(1)\rightarrow Retail(j)}$. A squared coefficient of variation (SCV), represented as $[c^2_{PM(1)\rightarrow Retail(j)}]_a$, is used to take into account the highly variable time between returns. Total available volume is bounded by $R_{PM(1)}$ from which a fraction $\varsigma_{PM(1)\rightarrow Retail(j)}$ is sent to retailer (j), or in equation form

$$\varsigma_{PM(1)\rightarrow Retail(j)} R_{PM(1)} = \lambda_{PM(1)\rightarrow Retail(j)}$$

The volume $R_{PM(1)}$ depends on the marginal value of time and the product lifetime. For a classification of products for end-of-life acquisition based on these criteria, we refer to Morana and Seuring (2007).

Each facility node has a capacity level (q), which is selected from a set of candidate levels. A retailer's total return rate when installed at capacity level (q) then becomes $\lambda_{Retail(j)(q)}$. So with $\sum_j \varsigma_{PM(1)\rightarrow Retail(j)} \leq 1$, we have

$$R_{PM(1)} \sum_j \varsigma_{PM(1)\rightarrow Retail(j)} = \sum_j \lambda_{PM(1)\rightarrow Retail(j)} = \sum_j \sum_q \lambda_{Retail(j)(q)}$$

Transportation towards the next network level, e.g. an evaluation centre, only starts when B units are collected at the retailer. This is the batch size that is going to be used at all forward and reverse network levels, so lot splitting and re-grouping of batches is not allowed. This would introduce additional flow disturbances that is beyond the scope of this study. The batch size is bounded by the maximum load of the transportation vehicles. The same reasoning and similar equations apply at the input of the forward chain where an aggregate source supplies raw materials from a single node $RM(1)$ to a manufacturing node $Mnfct(j)$.

In general, when all the operations at facility (i) of the current network level l' are finished, products are sent to a downstream facility (j) at a network level l for the next operations according to a rate $\lambda_{l'(i)\rightarrow l(j)}$. Total volume rate at the inbound of that facility (i) is $\lambda_{l'(i)}$. Since the product routing depends on the grade classification, a fraction $\omega_{l'\rightarrow l}$ is used to guide each product to the appropriate subsequent level in the network. Given the restriction of the model to a single product type, we assume that all facilities at a particular network level handle only one type of a product's grade. Processing different grades at a particular facility requires multiple, complex aggregations of arrival and process characteristics in the queueing model and would prevent clear insights in Section 5. As a result of that assumption, it does not matter to which specific facility the fraction $\omega_{l'\rightarrow l}$ is sent and we can omit the indices (i) and (j). This grade distribution is obviously applied at an evaluation centre where the quality is assessed: as-good-as-new products are re-distributed in the forward chain while inferior products continue in the reverse chain towards a single secondary market node $SM(1)$. However, we also use $\omega_{l'\rightarrow l}$ when the product flow between the two corresponding levels l' and l is not allowed (value is zero) or to dispose of units (value is $\omega_{l'\rightarrow Disp}$). Note that $\omega_{l'\rightarrow l}$ is a fixed, predefined parameter in the model, while $\varsigma_{l'(i)\rightarrow l(j)}$ is a fraction to decide on the percentage of the product flow that is assigned to each of the downstream facilities (j). Apart from binary decision variables to open or close facilities, these fractions are also to be decided on by

the model.

The way the disposals are handled in the model requires additional clarification. At the outbound of a facility (i) at network level l', a fraction $\omega_{l'\to Disp}$ is directed towards an aggregated external sink due to some technical limitations of the process (i.e. srap). The remaining number of units, which equals $(1-\omega_{l'\to Disp})\lambda_{l'(i)}$, is guided towards one or several transportation lines that link the facility with the appropriate processes (j) at downstream network levels l based on the product flow decisions $\varsigma_{l'(i)\to l(j)}$ and the grade distribution $\omega_{l'\to l}$ according to

$$\lambda_{l'(i)\to l(j)} = \varsigma_{l'(i)\to l(j)}\omega_{l'\to l}(1-\omega_{l'\to Disp})\lambda_{l'(i)} \tag{1}$$

When $\sum_l\sum_j \varsigma_{l'(i)\to l(j)}\omega_{l'\to l} \leq 1$, the constraint is satisfied that total output at facility (i) is less than or equal to its total input, excluding the disposals caused by technical issues. Since the disposal option can be considered as a single node at a separate layer l in the network, the model may decide to send there an additional fraction that equals $1-\sum_l\sum_j \varsigma_{l'(i)\to l(j)}\omega_{l'\to l}$. This means that the effective number of discarded products can be higher than what is technically expected from the process, not only to balance return and demand quantities, but also to use the model as a decision tool for the volume in the network that is optimal from a lead time perspective. Any excess products will create excess waiting time and excess work-in-process that both have a negative effect on the performance. The ASP model decides upon the optimal product flow that continues. Total disposal quantity equals

$$(\omega_{l'\to Disp} + (1-\sum_l\sum_j \varsigma_{l'(i)\to l(j)}\omega_{l'\to l}))\lambda_{l'(i)} \tag{2}$$

Based on Equation 1, total arrival rate at facility (j) becomes

$$\lambda_{l(j)} = \sum_{l'}\sum_i \lambda_{l'(i)\to l(j)}$$

Since each facility (j) can be installed at one capacity level (q) chosen from a set of candidate levels Q, we have

$$\lambda_{l(j)} = \sum_q \lambda_{l(j)(q)}$$

Because of the same batch size B at all network levels, the batch arrival rate at a transportation line between two facilities (i) and (j) and the batch arrival rate at a facility (j) respectively become

$$[\lambda_{l'(i)\to l(j)}]_{ba} = \lambda_{l'(i)\to l(j)}/B$$
$$[\lambda_{l(j)}]_{ba} = \lambda_{l(j)}/B$$

3.2. *Production Facility*

In this section, all the parameters are derived that are required to calculate total expected lead time of a product at facility (j), i.e. $EW_{l(j)}$. It consists of the following components

- Expected waiting time in the queue of batches $EWQ_{l(j)}$
- Expected process time of a batch, which equals $Bt_{l(j)}$

Binary variables $Y_{l(j)(q)}$ are used to select the appropriate capacity level (q) from a set of candidate levels Q at which a facility (j) at network level l needs to be installed. Given a maximum production quantity of $C_{l(j)(q)}$ units per time period at level (q), the mean 'effective' process time $t_{l(j)}$, which is the inverse of the mean 'effective' process rate $\mu_{l(j)}$, equals $1/\sum_q Y_{l(j)(q)} C_{l(j)(q)}$. By 'effective' we mean that it accounts for issues like setups, rework, breakdowns, etc. Note that in order to avoid additional aggregation steps, we assume that process times are independent of the origin of the product flow. This implies that each facility is able to process just one kind of the multiple grade classifications, which fits the single product type assumption. As long as the quality is not determined, which is not the case until after the evaluation centre, the product flow can also be considered as being of a single grade, i.e. the unclassified type.

We can formulate the facility utilisation as

$$\rho_{l(j)} = \lambda_{l(j)} t_{l(j)} = \frac{\lambda_{l(j)}}{\sum_q Y_{l(j)(q)} C_{l(j)(q)}} < 1 \tag{3}$$

Due to various and usually unknown customer usage patterns, much more variation in process times for returned products is observed in practice when compared to traditional forward supply chains. We describe this variability by its 'effective' SCV $[c^2_{l(j)}]_p$, which includes all sources of process disturbance. Since independent and identically distributed process times are a reasonable assumption (remanufacturing units can be modelled as an uncorrelated process with the same distribution), the SCV of the batch production time equals

$$[c^2_{l(j)}]_{bp} = [c^2_{l(j)}]_p / B \tag{4}$$

The variability of the expected batch interdeparture times at the outbound of facility (j) is found by the following linking equation (Whitt 1983)

$$[c^2_{l(j)}]_{bd} \approx (1 - \rho^2_{l(j)})[c^2_{l(j)}]_{ba} + \rho^2_{l(j)}[c^2_{l(j)}]_{bp} \tag{5}$$

An expression for the SCV of the batch interarrival time at facility (j), i.e. $[c^2_{l(j)}]_{ba}$, is based on the superposition of the products that are supplied by different transportation lines starting at upstream network levels. To obtain this aggregate SCV, we use the simplified approximation from Whitt (1983)

$$[c^2_{l(j)}]_{ba} \approx w_{l(j)} \sum_{l'} \sum_{i} \pi_{l'(i)\to l(j)} [c^2_{l'(i)\to l(j)}]_{bd} + 1 - w_{l(j)} \tag{6}$$

where

$$w_{l(j)} = 1/\left(1 + 4\left(1 - \rho_{l(j)}\right)^2\left(v_{l(j)} - 1\right)\right)$$
$$v_{l(j)} = 1/\left(\sum_{l'}\sum_{i}\pi^2_{l'(i)\to l(j)}\right)$$
$$\pi_{l'(i)\to l(j)} = [\lambda_{l'(i)\to l(j)}]_{ba}/[\lambda_{l(j)}]_{ba} \tag{7}$$

and $[c^2_{l'(i)\to l(j)}]_{bd}$ is equal to Equation (14) (see Section 3.3). Equation (7) represents the probability that the inbound flow at facility (j) comes from the transportation line under consideration.

Now we are able to estimate total expected lead time at the facilities. We opt to use the approximation from Whitt (1993) because it estimates the mean waiting time under heavy traffic conditions more accurately due to a correction factor ϕ. We refer to Whitt (1993) for more details about this factor. Using Equations (3), (4) and (6), we can formulate the expected waiting time of batches in the queue at facility (j) as

$$EWQ_{l(j)} \approx \phi(\rho_{l(j)}, [c^2_{l(j)}]_{ba}, [c^2_{l(j)}]_{bp}, 1)\left(\frac{[c^2_{l(j)}]_{ba} + [c^2_{l(j)}]_{bp}}{2}\right)\left(\frac{\rho_{l(j)}}{1-\rho_{l(j)}}\right)Bt_{l(j)}$$

Note that each individual unit of the batch experiences this expected waiting time and that each facility is modelled as a queueing system with a single machine in steady state. Little's law can then be used to find the expected number of products at facility (j), which will be used to calculate inventory costs

$$WIP_{l(j)} \approx \lambda_{l(j)}EW_{l(j)} \approx \lambda_{l(j)}(EWQ_{l(j)} + Bt_{l(j)}) \tag{8}$$

3.3. *Transportation Line*

An expected 'effective' transportation time $t_{l'(i)\to l(j)}$ with a SCV $[c^2_{l'(i)\to l(j)}]_t$ are used to include all disturbing factors like vehicle breakdowns, loading and unloading, traffic jams, etc. when shipping goods between two locations (i) and (j). We assume one transportation mode, e.g. trucks, and we ignore the transportation from the aggregate starting node $PM(1)$, i.e. between customer and retailer in the reverse chain, and from the other starting node $RM(1)$, i.e. between supplier and manufacturer in the forward chain. The reason is that the customer and the supplier are not considered to be part of the queueing system. Furthermore, $t_{l'(i)\to l(j)}$ is independent of the quantity being shipped. This feature indicates that we are dealing with a parallel batching process with batch size B, expressed in units (Hopp and Spearman 2000). Therefore, the total expected lead time of a product at the transportation line between locations (i) and (j), i.e. $EW_{l'(i)\to l(j)}$, is the sum of

- $WTBT_{l'(i)\to l(j)}$, the expected waiting time to form a batch of size B
- $EWQ_{l'(i)\to l(j)}$, the expected waiting time of batches in the queue
- $t_{l'(i)\to l(j)}$, the expected transportation time of the batch

Since a batch is not split as it proceeds through the network, there is no collection time required to form a batch of size B, except at the outbound of the initial level in

each network type, i.e. retailer nodes in the reverse chain and manufacturing nodes in the forward chain. These wait-to-batch-times (WTBT), which are added to $EW_{Retail(i)\rightarrow l(j)}$ and $EW_{Mnfct(i)\rightarrow l(j)}$ respectively, are expected to be equal to

$$WTBT_{Retail(i)\rightarrow l(j)} = \frac{B-1}{2\lambda_{Retail(i)\rightarrow l(j)}}$$

$$WTBT_{Mnfct(i)\rightarrow l(j)} = \frac{B-1}{2\lambda_{Mnfct(i)\rightarrow l(j)}}$$

This is because the first unit in a batch waits for $B-1$ other units to arrive and therefore waits $(B-1)/\lambda$ time units whereas the last one does not have to wait at all to join the batch. A similar approach can be found in Vandaele *et al.* (2003).

A stable system requires an effective utilisation level $\rho_{l'(i)\rightarrow l(j)}$ for each transportation line to be smaller than 1, or

$$\rho_{l'(i)\rightarrow l(j)} = \frac{[\lambda_{l'(i)\rightarrow l(j)}]_{ba} t_{l'(i)\rightarrow l(j)}}{m_{l'(i)\rightarrow l(j)}} < \rho^{Max}_{l'(i)\rightarrow l(j)} < 1 \tag{9}$$

By imposing an upper bound $\rho^{Max}_{l'(i)\rightarrow l(j)}$, we transform the total number of activated vehicles $m_{l'(i)\rightarrow l(j)}$ into a parameter with a fixed value, which avoids the introduction of additional decision variables. This upper bound is an input parameter for the model, and its value should be chosen in such a way that excess waiting times are avoided. Based on our experience, we suggest to set it at approximately 90%. Equation 9 implies the following lower bound on the number of vehicles

$$m_{l'(i)\rightarrow l(j)} = \frac{[\lambda_{l'(i)\rightarrow l(j)}]_{ba} t_{l'(i)\rightarrow l(j)}}{\rho^{Max}_{l'(i)\rightarrow l(j)}}$$

which must be rounded up to the nearest integer value according to

$$m_{l'(i)\rightarrow l(j)} = INT[m_{l'(i)\rightarrow l(j)} + 1] \tag{10}$$

With respect to the SCV of the batch interarrival time at a transportation line between locations (i) and (j), i.e. $[c^2_{l'(i)\rightarrow l(j)}]_{ba}$, we have to distinguish two situations. At the first level in the network, we add B IID interarrival times with a SCV of either $[c^2_{PM(1)\rightarrow Retail(j)}]_a$ at the retailer in the reverse chain or $[c^2_{RM(1)\rightarrow Mnfct(j)}]_a$ at the manufacturer in the forward chain, resulting in

$$[c^2_{PM(1)\rightarrow Retail(j)}]_{ba} \approx [c^2_{PM(1)\rightarrow Retail(j)}]_a/B \tag{11}$$

$$[c^2_{RM(1)\rightarrow Mnfct(j)}]_{ba} \approx [c^2_{RM(1)\rightarrow Mnfct(j)}]_a/B \tag{12}$$

On the other hand, for product flows between any two other locations further down the network, we have to use another equation because the variability is changed by the starting facility (i) at network level l', which was an ending node (j) at an upstream network level l. To this end, we first need the variability in departures from that facility, which was found in Equation 5. So we have $[c^2_{l(j)}]_{bd} = [c^2_{l'(i)}]_{bd}$. Next, the product stream that leaves this facility may be split over various transportation lines because of multiple

destinations. This complicates the handling of the SCV's, even though superposition of product flows must not be performed because only one facility node (i) feeds this line. The relation between $[c^2_{l'(i)\to l(j)}]_{ba}$ and $[c^2_{l'(i)}]_{bd}$ has been proven in the literature (Shanthikumar and Buzacott 1981, Lambrecht *et al.* 1998) and is given by

$$[c^2_{l'(i)\to l(j)}]_{ba} \approx \pi_{l'(i)\to l(j)}[c^2_{l'(i)}]_{bd} + (1 - \pi_{l'(i)\to l(j)}) \tag{13}$$

with $\pi_{l'(i)\to l(j)}$ the proportion of processed products at the current facility (i) that is going to destination node (j) and equals $= [\lambda_{l'(i)\to l(j)}]_{ba}/[\lambda_{l'(i)}]_{ba}$.

The final variability measure that needs to be derived, is the variability of the departures when products exit the transportation line. Although Buzacott and Shanthikumar (1993) have presented several approximations for systems with multiple servers, for our purposes it is reasonable to estimate it as (Hopp and Spearman 2000)

$$\begin{aligned}[c^2_{l'(i)\to l(j)}]_{bd} \approx 1 + (1 - \rho^2_{l'(i)\to l(j)})([c^2_{l'(i)\to l(j)}]_{ba} - 1) + \\ \frac{\rho^2_{l'(i)\to l(j)}}{\sqrt{m_{l'(i)\to l(j)}}}([c^2_{l'(i)\to l(j)}]_t - 1)\end{aligned} \tag{14}$$

Using Equations (9), (10), (11), (12) and (13), we can formulate the mean unit waiting time in the queue at each transportation line between locations (i) and (j) according to the approximation from Whitt (1993) as

$$\begin{aligned}EWQ_{l'(i)\to l(j)} \approx \phi(\rho_{l'(i)\to l(j)}, [c^2_{l'(i)\to l(j)}]_{ba}, [c^2_{l'(i)\to l(j)}]_t, m_{l'(i)\to l(j)}) * \\ \left(\frac{[c^2_{l'(i)\to l(j)}]_{ba} + [c^2_{l'(i)\to l(j)}]_t}{2}\right) * \\ \left(\frac{\rho^{(\sqrt{2(m_{l'(i)\to l(j)}+1)}-1)}_{l'(i)\to l(j)}}{m_{l'(i)\to l(j)}(1 - \rho_{l'(i)\to l(j)})}\right) t_{l'(i)\to l(j)}\end{aligned}$$

Little's law can then be used to find the expected number of products at each transportation line between locations (i) and (j), which will be used to calculate inventory costs

$$\begin{aligned}WIP_{l'(i)\to l(j)} &\approx \lambda_{l'(i)\to l(j)} EW_{l'(i)\to l(j)} \\ &\approx \lambda_{l'(i)\to l(j)}(WTBT_{l'(i)\to l(j)} + EWQ_{l'(i)\to l(j)} + t_{l'(i)\to l(j)})\end{aligned} \tag{15}$$

3.4. *Objective Function and Constraints*

Our reverse logistics network design model becomes

Max

$$pr_{PM(1)}\sum_i \lambda_{Retail(i)\to PM(1)} + pr_{SM(1)}\sum_i \lambda_{Retail(i)\to SM(1)} - \qquad \text{REV}$$

$$\sum_l\sum_j\sum_q FIX_{l(j)(q)}Y_{l(j)(q)} - \qquad \text{FC}$$

$$\sum_{l}\sum_{j}\sum_{q} vc_{l(j)(q)}\lambda_{l(j)(q)} - \qquad \text{VC}$$

$$\sum_{l'}\sum_{l}\sum_{i}\sum_{j} tc_{l'(i)\to l(j)} d_{l'(i)\to l(j)} [\lambda_{l'(i)\to l(j)}]_{ba} - \qquad \text{TC}$$

$$\sum_{l'}\sum_{l}\sum_{i}\sum_{j} hc_{l'(i)\to l(j)} WIP_{l'(i)\to l(j)} - \sum_{l}\sum_{j} hc_{l(j)} WIP_{l(j)} - \qquad \text{IC}$$

$$\sum_{l'}\sum_{i} dc_{l'(i)} \left(\lambda_{l'(i)} - \sum_{l}\sum_{j} \lambda_{l'(i)\to l(j)} \right) - \qquad \text{DC}$$

$$pcr_{PM(1)} \left(1 - \sum_{j} \varsigma_{PM(1)\to Retail(j)} \right) R_{PM(1)} - \qquad \text{PCR}$$

$$pcd_{PM(1)} \left(D_{PM(1)} - \sum_{i} \lambda_{Retail(i)\to PM(1)} \right) -$$

$$pcd_{SM(1)} \left(D_{SM(1)} - \sum_{i} \lambda_{Retail(i)\to SM(1)} \right) \qquad \text{PCD} \qquad (16)$$

subject to

Balance constraints

$$\begin{aligned}
&Input && R_{PM(1)} \sum_{j} \varsigma_{PM(1)\to Retail(j)} = \sum_{j} \lambda_{PM(1)\to Retail(j)} \\
& && R_{RM(1)} \sum_{j} \varsigma_{RM(1)\to Mnfct(j)} = \sum_{j} \lambda_{RM(1)\to Mnfct(j)} \\
&Facility \quad \forall l \quad \forall j && \lambda_{l(j)} = \sum_{q} \lambda_{l(j)(q)} \\
& && \quad = \sum_{l'}\sum_{i} \lambda_{l'(i)\to l(j)} \\
& && \quad = \sum_{l'}\sum_{i} \varsigma_{l'(i)\to l(j)} \omega_{l'\to l} (1 - \omega_{l'\to Disp}) \lambda_{l'(i)} \\
&Demand && \sum_{i} \lambda_{Retail(i)\to PM(1)} \leq D_{PM(1)} \\
& && \sum_{i} \lambda_{Retail(i)\to SM(1)} \leq D_{SM(1)} \\
&Fractions \; \forall l' \quad \forall i && \sum_{l}\sum_{j} \varsigma_{l'(i)\to l(j)} \omega_{l'\to l} \leq 1
\end{aligned}$$

Capacity constraints

$$\begin{aligned}
&\forall l \quad \forall j \quad \forall q \qquad \lambda_{l(j)(q)} \leq C_{l(j)(q)} Y_{l(j)(q)} \\
&\forall l \quad \forall j \quad \forall q \qquad \lambda_{l(j)(q)} \geq C_{l(j)(q-1)} Y_{l(j)(q)} \\
&\forall l \quad \forall j \qquad \sum_{q} Y_{l(j)(q)} \leq 1
\end{aligned}$$

Logical constraints

$$\forall l \;\; \forall j \qquad Y_{l(j)(q)} = \{0,1\}$$
$$\forall l \;\; \forall l' \;\; \forall i \;\; \forall j \; 0 \leq \varsigma_{l'(i)\rightarrow l(j)} \leq 1$$

Apart from revenue (REV) that is generated when products leaving the retailer are sold at some unit price $pr_{l(1)}$ in either the primary ($l = PM$) or secondary market ($l = SM$), the objective function consists of different types of costs. First of all, a capacity dependent fixed cost $FIX_{l(j)(q)}$ is charged for each facility that is opened (FC). It accounts for periodic costs like overhead, depreciation, interest, etc. Another cost component is the variable processing cost (VC) that is correlated with the magnitude of the product flow. The unit production cost $vc_{l(j)(q)}$ for e.g. material, labor and energy costs vary with the size of the plant. Initially, this cost may decline as a function of capacity because of economies of scale. This is due to lower operating costs and efficiency gains, possibly reinforced by learning effects. However, at some point, the opposite behavior occurs because the plant becomes too large to be efficient (diseconomies of scale). Total transportation cost (TC) is assumed to be a linear function of total distance travelled during the specific period, where the distance unit cost is denoted by $tc_{l'(i)\rightarrow l(j)}$. Furthermore, the batch size determines how often the distance $d_{l'(i)\rightarrow l(j)}$ between two nodes is covered. The number of shipments equals total flow rate divided by the truck load, i.e. $[\lambda_{l'(i)\rightarrow l(j)}]_{ba}$. So TC is proportional to the volume too. Next, we have the inventory costs (IC) associated with the work-in-process levels derived in Equations 8 and 15 for which it costs respectively $hc_{l(j)}$ and $hc_{l'(i)\rightarrow l(j)}$ to hold one unit in stock during the period under consideration. Products that leave the system at a facility (i) at network level l' are charged a unit disposal cost of $dc_{l'(i)}$ resulting in an overall disposal cost (DC). Total disposal quantity is similar to Equation 2. Finally, a penalty cost is incurred for the returned quantity that is not allowed from the primary market into the network (PCR) and for the demand quantity that is not satisfied in each market type (PCD).

Several constraints are required to obtain feasible solutions. Main constraints are the various balance equations to control the product flow assignments. To begin with, both input constraints ensure that all products, either returned from the primary market or delivered from the raw material supply, are bounded by their respective maximum values of $R_{PM(1)}$ and $R_{RM(1)}$. This is guaranteed by the general constraint concerning the fractions where $\omega_{PM\rightarrow Retail} = 1$ and $\omega_{RM\rightarrow Retail} = 1$ at the initial network level because of a single quality level upon arrival. This constraint regarding the fractions ensures that a facility's output will not exceed its input. These fractions are applied to the net production quantity, i.e. total input reduced by the disposal fraction (see the facility constraint). This means that when the fractional constraint is less than 100%, more units than the expected scrap leaves the system. This is useful when total supply exceeds demand or downstream capacities, or when a faster response through less volume is required. Concerning the facility constraints, we aggregate the production quantities at all capacity levels (q) in a facility (j) to obtain its overall production quantity $\lambda_{l(j)}$. This value, which depends on all the feeding transportation lines, includes the units that are discarded, or in other words, the quantity that leaves the queueing system is determined after processing. The units that continue to downstream stations are controlled by the scrap fraction, the predefined grade distribution and the product flow assignment decision. We complete the discussion of the balance constraints with the bounds on consumer market demand. Another constraint category is dedicated to capacity limits. Total production quantity in facility (j) when installed at capacity level

(q) should not exceed its upper bound, but at the same time should not go below its lower bound, which equals the upper bound of the previous capacity level $(q-1)$. Note that $C_{l(j)(0)} = 0$. The last capacity constraint forces the model to open only one capacity level for each facility. Finally, there are some obvious logical constraints for binary and fractional decision variables.

This objective function is defined for a steady state period. If the time horizon is e.g. one year, all parameters must reflect this: the fixed cost being the overhead cost to keep the operations running for one year, time parameters being expressed in years, rate parameters being expressed per year, the holding cost being the average cost to hold one unit in stock for one year, etc.

We can learn from Equation 16 that nonlinear queueing relationships are combined with continuous and binary decision variables. As a result, the proposed model belongs to the class of Mixed Integer Non-Linear problems (MINLP). These problems are difficult to solve because they combine all the difficulties of two sub-problems that are both NP-complete: the combinatorial nature of mixed integer programs and the difficulty in solving nonlinear programs. Like for MILP problems, the computational complexity grows exponentially with the number of discrete variables and the number of decisions within each discrete variable (open or close), but in addition to this also with the number of all variables entering nonlinearly into the model. This requires an advanced algorithm, which is described in the next section.

4. Model Optimization

The continuous values for the fractions that assign the product flow to downstream facilities (i.e. $\varsigma_{l'(i)\rightarrow l(j)}$) and the binary values to open or close a facility (i.e. $Y_{l(j)(q)}$) are the decision variables in the ASP model. In order to determine their solution value, we apply the Differential Evolution (DE) heuristic, a member of the broader family of Genetic Algorithms. Since this approach is similar to Lieckens and Vandaele (2007), only the major steps of DE are briefly outlined. In comparison with that study, the ASP model developed here is to be applied to much larger, more realistic situations. To that end, we optimize fractions assigned to a facility, instead of absolute values for the flow at a facility's inbound and outbound. This approach limits the number of constraints because the additional check that its output does not exceed its input can be removed. Another improvement is the alternative method to handle all the constraints in the model. In this section, we prefer to focus on that method.

The DE algorithm is an appropriate solution method for our network design problem because the following characteristics, which are listed by Storn and Price (1997), apply

- it is simple, fast and robust;
- it has a superior global optimization ability;
- it can easily be implemented in a parallel computing environment, to speed up the optimization;
- it is effective in nonlinear optimization and can be very easily adapted for mixed parameter optimization;
- it handles undifferentiable objective functions;
- it operates on flat surfaces;
- it can provide multiple solutions in a single run.

In addition, after studying seven difficult design and control MINLP problems in

chemical engineering, Babu and Angira (2002) conclude that the technique of DE is the best evolutionary computation method. According to Lampinen and Zelinka (1999), DE outperforms any of the competing methods like branch & bound using sequential quadratic programming, integer-discrete-continuous nonlinear programming, simulated annealing, genetic algorithm, nonlinear mixed-discrete programming, ..., also for difficult nonlinear objective functions with multiple non-trivial constraints.

As an improved version of genetic algorithms, DE belongs to the class of evolutionary algorithms (EA) that are based on the principle of survival of the fittest. It is basically a computerised, population based search and optimization algorithm that differs from EA's in the way the mutation is driven. Instead of using the output of a predefined distribution function, DE uses the difference of randomly sampled pairs of object vectors. Their response to the objective function determines their distribution, which on its turn determines the distribution of the object vector differences. So the mutation that improves the object vectors reflects information of the objective function it is optimizing. Instead of using only local information for each object vector, DE mutates all object vectors with the same universal distribution. In this way, there is a higher chance to cover the whole search space and to find a global optimum.

The method is defined as a parallel direct search method which operates on a population P_G of constant size that is associated with each generation G and consists of NP vectors, or candidate solutions, $\vec{X}_{p,G}$ with $p = 1, 2, \ldots, NP$. Each vector $\vec{X}_{p,G}$ consists of D decision variables $X_{o,p,G}$ with $o = 1, 2, \ldots, D$. This is briefly summarised as

$$\begin{aligned} P_G &= \left\{\vec{X}_{1,G}, \vec{X}_{2,G}, \ldots, \vec{X}_{p,G}, \ldots, \vec{X}_{NP,G}\right\}, \\ \vec{X}_{p,G} &= \{X_{1,p,G}, X_{2,p,G}, \ldots, X_{o,p,G}, \ldots, X_{D,p,G}\}, \\ G &= 1, \ldots, G_{\max}, \\ NP &\geq 4. \end{aligned}$$

Each fractional product flow assignment and each binary value to open or close a facility are then considered as the decision variables, $\varsigma_{l'(i)\to l(j)} \equiv X_{o,p,G}$ and $Y_{l(j)(q)} \equiv X_{o,p,G}$. Note that we can omit the decision variable $\varsigma_{l'(i)\to l(j)}$ when $\omega_{l'\to l} = 0$. This is important from a computational perspective because it limits the value of D for this NP-hard problem.

The different steps of the DE algorithm as used and described in more detail in Lieckens and Vandaele (2007) are now listed.

Step 1: Choose a strategy. Although many different schemes exist (Storn and Price 1997), we only use the *DE/rand/1/bin* scheme in the application in Section 5. See below for more details.

Step 2: Initialise the key parameters of control. The user-defined control parameters, which remain constant during the search process, are the crossover constant CR, the population size NP, the mutation scaling factor F, the coefficient of combination K and the maximum number of generations $G_{\max}$.

Step 3: Initialise the population. The initial population $P_{G=0}$ provides us with a starting solution for optimum seeking and is chosen randomly within the bounds of the parameters that are set by the constraints. It should cover the entire variable space. A population for $G \geq 0$ is allowed to contain members that have infeasible

values for the decision variables with respect to the model constraints (see below in this section), but not with respect to their specific upper and lower bounds (see Step 6).

Step 4: Evaluate the profit of each vector in the population and find the one with the highest profit.

Step 5: Perform mutation and recombination. Mutation aims to keep a population robust and to search a new area. In order to move existing object vectors $\vec{X}_{p,G}$ in the right direction by the right amount at the right time, DE adds the weighted difference of randomly sampled pairs of vectors in the current population P_G until a mutated vector $\vec{V}_{p,G+1}$ is obtained. In order to reinforce prior successes, a recombination or crossover operation is required as a complementary step to mutation. It creates a trial vector $\vec{U}_{p,G+1}$ by selecting elements from the current target vector $\vec{X}_{p,G}$ and the mutated donor vector $\vec{V}_{p,G+1}$. The crossover constant CR controls the probability that a trial vector element will come from the mutated vector $\vec{V}_{p,G+1}$, instead of from the current vector $\vec{X}_{p,G}$, and therefore ranges from 0 to 1.

Step 6: Check lower and upper bounds of the variables. The parameters of the child vectors must be checked for boundary conditions. In the ASP model, these bounds correspond to the logical constraints in Section 3.4. If a mutated parameter exceeds such bound, we set it equal to the middle of its original parent value and that boundary value.

Step 7: Perform selection. To select the vectors for the next generation, each child has to be evaluated by the objective function and compared with its parent's objective value. If the profit of the child is greater than or equal to the profit of its parent, it replaces that parent in the population, otherwise the parent will be retained in the next generation. As a result, all the individuals of the next generation are as good as or better than their counterparts in the current generation.

Step 8: Repeat the evolutionary cycle until $G_{\max}$ is reached, or even better, until all vectors are converged to a single solution with precision ϵ_1. This is true when $\left|\frac{\text{Best member - Worst member}}{\text{Worst member}}\right| < \epsilon_1$. The smaller the ϵ_1-value, the longer the algorithm will search for a solution, but the higher its quality.

In the *DE/rand/1/bin* scheme, the population of child or trial vectors $P'_{G+1} = \vec{U}_{p,G+1} = U_{o,p,G+1}$ for each parent or target vector $Q_{o,p,G}$ is created as follows

$$U_{o,p,G+1} = \begin{cases} V_{o,p,G+1} = Q_{o,r_3,G} + F\left(Q_{o,r_1,G} - Q_{o,r_2,G}\right) & \text{if } R(0) \leq CR \vee o = k \\ Q_{o,p,G} & \text{otherwise} \end{cases}$$

where

- $p = 1, ..., NP > 3, o = 1, ..., D$
- $k \in \{1, ..., D\}$, random decision variable index
- $r_1, r_2, r_3 \in \{1, 2, ..., NP\}$ randomly selected, but $r_1 \neq r_2 \neq r_3 \neq p$
- $CR \in [0, 1]$, $F \in (0, 1+]$, and $R(0) \in [0, 1]$ is a uniformly random number

The /1/ in this scheme means that there is one paired difference of randomly (/rand/) chosen vectors, weighted with F, that drives the mutation. The randomly chosen indices r_1, r_2 and r_3 must be mutually different, and different from the current parent object vector p. Consequently, NP must be greater than 3. Other, random integer values for r_1, r_2 and r_3 are chosen for each individual candidate solution p. The index k ensures that each child vector will differ from its parent in the previous generation by at least

one variable. A new random integer value is assigned to k prior to the construction of each child vector. The binomial scheme ($/bin/$) takes parameters from $\vec{V}_{p,G+1}$ each time when $R(0) \leq CR$, otherwise the parameters come from $\vec{Q}_{p,G}$. We refer to Storn and Price (1997), Lampinen and Zelinka (1999), and Lin *et al.* (1999) for more details about the mutation schemes, values for the control parameters, and other stopping criteria.

Constraints limit the feasible solutions to a subset of the total search space. A popular approach is to implement them as 'soft' constraints (Michalewicz and Schoenauer 1996). This means that penalty functions cause an object vector's profit to decrease with both the magnitude and number of its constraint violations. In this way, the problem is converted into an unconstrained problem by penalising the infeasible solutions. Each constraint is reformulated in such a way that it is greater than zero when it is violated.

The main problem with the penalty function approach is finding appropriate penalty values, clearly when there are many constraints (Powell 1978). Large penalties may result in both infinite objective values, which are difficult to handle by computers, and high convergence rates, which lead to suboptimal solutions. On the other hand, low penalties may result in either a slow convergence towards feasible solutions or even worse infeasible solutions because the constraint violation does not contribute a high cost to the penalty function. Finding an appropriate setting is time consuming because it requires multiple runs, each time adjusting the penalty parameters by trial-and-error. Therefore, we prefer to use the alternative constraint handling method from Lampinen (2001), which is summarised as

$$\vec{X}_{p,G+1} = \begin{cases} \vec{U}_{p,G+1} \text{ if } \begin{cases} \begin{cases} \forall m \in \{1,2,...,\upsilon\} : g_m\left(\vec{U}_{p,G+1}\right) \leq 0 \wedge g_m\left(\vec{X}_{p,G}\right) \leq 0 \\ \wedge \\ f\left(\vec{U}_{p,G+1}\right) \geq f\left(\vec{X}_{p,G}\right) \end{cases} \\ \vee \\ \begin{cases} \forall m \in \{1,2,...,\upsilon\} : g_m\left(\vec{U}_{p,G+1}\right) \leq 0 \\ \wedge \\ \exists m \in \{1,2,...,\upsilon\} : g_m\left(\vec{X}_{p,G}\right) > 0 \end{cases} \\ \vee \\ \begin{cases} \exists m \in \{1,2,...,\upsilon\} : g_m\left(\vec{U}_{p,G+1}\right) > 0 \\ \wedge \\ \forall m \in \{1,2,...,\upsilon\} : \\ Max\left[g_m\left(\vec{U}_{p,G+1}\right),0\right] \leq Max\left[g_m\left(\vec{X}_{p,G}\right),0\right] \end{cases} \end{cases} \\ \vec{X}_{p,G} \quad \text{otherwise} \end{cases}$$

where $g_m\left(\vec{Z}\right)$ represents the constraint for a vector $\vec{Z}$. The trial vector $\vec{U}_{p,G+1}$ when compared with the current population member $\vec{X}_{p,G}$ will be selected if

- all constraints for $\vec{U}_{p,G+1}$ and $\vec{X}_{p,G}$ are satisfied, but $\vec{U}_{p,G+1}$ has a better profit value
- all constraints for $\vec{U}_{p,G+1}$ are satisfied but not for $\vec{X}_{p,G}$
- not all constraints for $\vec{U}_{p,G+1}$ are satisfied, but it violates all the constraints equal or less than $\vec{X}_{p,G}$

Note that the objective function is not evaluated until all the constraints are feasible, which reduces computation time and results in a fast convergence towards the feasible

regions of the search space. In addition, the algorithm will not always have to go through all the constraints, which further decrease computation time. This is interesting when computationally expensive objective and constraint functions are present. Lampinen reports for highly constrained problems that the first feasible solution is found 95%-99% faster than random search. Furthermore, 1000 independent optimization runs provided the same optimum solution that was equal or better than results found in the literature with other methods.

According to the procedure above, the constraints in our model are handled as follows

$$
\begin{array}{lll}
 & & \sum_i \lambda_{Retail(i)\rightarrow PM(1)} - D_{PM(1)} \leq 0 \\
 & & \sum_i \lambda_{Retail(i)\rightarrow SM(1)} - D_{SM(1)} \leq 0 \\
\forall l' & \forall i & \sum_l \sum_j \varsigma_{l'(i)\rightarrow l(j)} \omega_{l'\rightarrow l} - 1 \leq 0 \\
\forall l & \forall j \;\; \forall q & \lambda_{l(j)(q)} - C_{l(j)(q)} Y_{l(j)(q)} \leq 0 \\
\forall l & \forall j \;\; \forall q & C_{l(j)(q-1)} Y_{l(j)(q)} - \lambda_{l(j)(q)} \leq 0 \\
\forall l & \forall j & \sum_q Y_{l(j)(q)} - 1 \leq 0
\end{array}
$$

5. Model Application

The ASP model is now applied to an examplary, but realistic case study. The structure is based on the CLSC of Hewlett-Packard in the US printer industry (Guide *et al.* 2006), but the number of nodes is increased and the data is manipulated. The reason is that we want to verify whether or not the proposed optimization procedure with the adapted DE constraint handling method in combination with fractional flow assignments still efficiently performs for large real world problems that are modelled according to the relationships developed in Section 3. We prefer not to gain managerial insight based on a sensitivity analysis of various parameters. Instead, we formulate some scenarios for which good solutions can be proposed in advance. Comparison with the DE result will confirm the high performance of the algorithm.

5.1. *Data*

The CLSC structure for the application can be found in Figure 1, except that multiple potential locations may occur. The data for the scenarios below are based on Tables 1 to 4. Parameter dimensions are as follows: financial parameters are in euro, production capacities are in units per year, distances are in kilometers and transportation times are in the number of eight-hour-working days. Fixed costs are initially set to zero, while disposal, penalty and unit transportation costs are always zero.

In the reverse chain, maximum 3000 units return each year, of which 10% is discarded after evaluation and 1/3 can be resold as-new. The maximum demand for low grade products in the secondary market equals the maximum volume with this low quality level that can leave the system, or $(1 - 0.1) \times 2/3 \times 3000 = 1,800$ units. In the forward chain, the maximum amount of raw material supply is set equal to the sum of the yearly net demand for new products (30000) plus the yearly returned amount (3000). The reason is that we want the model to decide about the best mix of new and as-new products to fulfill the total gross demand of 33000 per year.

Two capacity levels are possible when multiple facility nodes exist for a specific process.

The first capacity level always equals the maximum product flow that can possibly enter the respective network level, which is based on maximum input quantities, minimum disposal fractions and routing probabilities. The second capacity level is this maximum product flow divided by the number of facility nodes at that network level.

Since it is mainly the WTBT and not the process of taking back products that determines the delays at the retailer, an infinite capacity is assumed here. We assume a batch size of 50 for all vehicles, exponentially distributed transportation times and no transportation from and to the input/output market. The use of highly variable return arrivals (SCV=2) and more stable raw material supplies (SCV=0.5) indicates the difference in variability between both chains. Also, the variability of the process times is supposed to be high in the reverse chain (SCV=1.5) and low in the forward chain (SCV=0.25).

5.2. *Scenario Analysis*

In this section, four scenarios are solved by DE and compared with an as good as possible solution that we suggest based on insight in the structure of the case. These solutions are referred to as 'quick and dirty' solutions (QDS). Table 6 presents their performance. We used the DE/rand/1/bin scheme with the following parameters: $F = 0.6$, $CR = 0.99$, $\epsilon = 10^{-7}$ and $NP = 4 \times D$, with $D = 122$. Table 5 summarises the computation effort on an Intel Core Duo Processor T2400, 1.83GHz. Note that we initialise the fractions in the population by using an upper bound of one divided by the number of facility nodes at the downstream network level. From the second generation on, the upper bound is increased to one. In this way, the whole search space is covered, initial solutions are feasible and fast convergence to local optima is avoided.

Scenario 1 is based on Tables 1 to 4 with $q = 1$ for all facilities, but with $t_{l'(i)\rightarrow l(j)} = 1, \forall l', l, i, j$. This means that all facilities are equally preferred, so a good QDS is a network where the total product flow is evenly distributed over all locations. From the results in Tables 6 and 7, we can conclude that even though the same revenue is generated, the DE algorithm is able to find an even better solution by re-assigning the product flows in such a way that less inventory costs are incurred. Evaluation of the objective function for three other open locations with the same volumes learns that alternative, but similar network layouts have the same profit. In Scenario 2, transportation times from Table 4 are used. This means that faster transportation lines should be preferred, so a good QDS is a network where the product flow assignments are adapted accordingly. For this solution, we applied fractions $\varsigma_{l'(i)\rightarrow l(j)}$ taking into account the relative share of the transportation times. From the results in Tables 6 and 8, we can conclude that less product flow is assigned to facility locations with a higher index (i.e. longer transportation time), and that DE finds a better network, not only by changing the layout, but also by reducing the input. Scenario 3 is the same as Scenario 1, but with $q = 2$ in case of multiple facilities. For QDS, we opt for the same input found by DE for this scenario 2, but with the product flow distribution alike the QDS in Scenario 1. From the results in Tables 6 and 9, we can conclude that the optimum volume in the network is reduced significantly, and that the product flow distribution according to DE is better due to less inventory costs. Scenario 4 is the same as Scenario 1 but with fixed costs as in Table 10. These values are chosen in such a way that the DE network layout from Scenario 1 yields a profit at approximately 1/3 of its original profit (see Table 6). That layout is an evident QDS. A better solution according to DE is to close the reverse chain and to sell less products in the forward chain (i.e. 31237 units). It is better to send this lower volume through only

one distribution centre, even though this strategy generates more inventory costs and less revenue, but it mainly saves on fixed costs. For all these scenarios, various sensitivities for minor deviations in the product flow and for other open location combinations have been performed nearby the DE solution, but none of them reveals a better network.

6. Conclusion

An Advanced Strategic Planning model for the design of complex Closed-Loop Supply Chains with multiple levels and a high degree of uncertainty is developed by integrating queueing relationships that measure the impact of delays and inventory levels. This combinatorial and nonlinear problem is difficult to solve. Different scenarios on a realistic, but structured problem for which good solutions can be defined in advance show that the Differential Evolution algorithm is a very powerful optimization tool. It outperforms these solutions by adjusting the location, re-assigning the product flow and reducing the input quantities, which proofs that the trade-offs in the model are perfectly handled. Furthermore, its stochastic search procedure may lead to alternative solutions that differ from those found by deterministic methods.

References

Babu, B. and Angira, R., 2002. A Differential Evolution Approach for Global Optimization of MINLP Problems. *Proceedings of 4th Asia-Pacific Conference on Simulated Evolution And Learning*, 2, 880–884.

Buzacott, J. and Shanthikumar, J., 1993. *Stochastic Models of Manufacturing Systems*. Englewood Cliffs, NY: Prentice Hall.

Fleischmann, M., van Nunen, J., and Graeve, B., 2003. Integrating Closed-Loop Supply Chains and Spare-Parts Management at IBM. *Interfaces*, 33 (6).

Guide, V.D.R., *et al.*, 2006. Time Value of Commercial Product Returns. *Management Science*, 52 (8), 1200–1214.

Hopp, W. and Spearman, M., 2000. *Factory Physics*. New York: The McGraw-Hill Companies.

Lambrecht, M., Ivens, P., and Vandaele, N., 1998. ACLIPS: A Capacity and Lead Time Integrated Procedure for Scheduling. *Management Science*, 44 (11), 1548–1561.

Lampinen, J. and Zelinka, I., 1999. 8. *In*: *Mechanical Engineering Design Optimization by Differential Evolution.*, 127–146 London: McGraw-Hill.

Lampinen, J., 2001. Multi-Constrained Nonlinear Optimization by the Differential Evolution Algorithm. *In*: *6th On-Line World Conference on Soft Computing in Industrial Applications (WSC6)*, September 10-24.

Lieckens, K. and Vandaele, N., 2007. Reverse logistics network design with stochastic lead times. *Computers & Operations Research*, 34, 395–416.

Lin, Y.C., Wang, F.S., and Hwang, K.S., 1999. A Hybrid Method of Evolutionary Algorithms for Mixed Integer Nonlinear Optimization Problems. *In*: *Proceedings of the 1999 Congress on Evolutionary Computation*, Vol. 3 of *Proceedings of the 2002 IEEE World Congress on Computation Intelligence* Piscataway, NJ, USA: IEEE, 2159–2166.

Michalewicz, Z. and Schoenauer, M., 1996. Evolutionary Algorithms for Constrained Parameter Optimization Problems. *Evolutionary Computation*, 4, 1–32.

Morana, R. and Seuring, S., 2007. End-of-life returns of long-lived products from end customer - insights from an ideally set up closed-loop supply chain. *International Journal of Production Research*, 45 (18), 4423–4437.

Powell, M., 1978. Algorithms for Nonlinear Constraints That Use Lagrangian Functions. *Mathematical Programming*, 14, 224–248.

Rubio, S., Chamorro, A., and Miranda, F., 2008. Characteristics of the research on reverse logistics (1995-2005). *International Journal of Production Research*, 46 (4), 1099–1120.

Shanthikumar, J. and Buzacott, J., 1981. Open Queueing Network Models of Dynamic Job Shops. *International Journal of Production Research*, 19 (3), 255–266.

Storn, R. and Price, K., 1997. Differential Evolution - a Simple End Efficient Heuristic for Global Optimization over Continuous Spaces. *Journal of Global Optimization*, 11, 341–359.

Vandaele, N., Vannieuwenhuyse, I., and Cupers, S., 2003. Optimal Grouping for a Nuclear Magnetic Resonance Scanner by Means of an Open Queueing Model. *European Journal of Operational Research*, 151 (1), 181–192.

Whitt, W., 1983. The Queueing Network Analyzer. *The Bell System Technical Journal*, 62 (9), 2779–2815.

Whitt, W., 1993. Approximations for the GI/G/m Queue. *Production and Operations Management*, 2 (2), 114–161.

Appendix A. Glossary

Table 1. Facility data.

l	j	q	$C_{l(j)(q)}$	$[c^2_{l(j)}]_p$	$vc_{l(j)(q)}$	$\omega_{l\rightarrow Disp}$	$hc_{l(j)}$
Retail	1	1	∞	1	0	0	0
Eval	1, 2, ..., 9	1	3000	1.5	1.5	10%	0.25
		2	333.33	1.5	1.5	10%	2.5
Rem	1, 2, ..., 6	1	1800	1.5	2.5	0	0.25
		2	300	1.5	2.5	0	0.25
Mnfct	1	1	33000	0.25	50	0	1.25
Distr	1, 2, 3	1	33000	0.25	20	0	1.25
		2	11000	0.25	20	0	1.25

Table 2. Input/Output data.

		Input		*Output*	
l'	l	$R_{l'(1)}$	$[c^2_{l'(1)\to l(1)}]_a$	$D_{l'(1)}$	$pr_{l'(1)}$
PM	*Retail*	33000	0.5	33000	200
SM	*Retail*	3000	2	1800	150

Table 3. Transportation data $\forall l', l, i, j$.

$\rho^{Max}_{l'(i)\rightarrow l(j)}$	90%
$[c^2_{l'(i)\rightarrow l(j)}]_t$	1
B	50
$hc_{l'(i)\rightarrow l(j)}$	0.25 (Reverse) - 1.25 (Forward)

Table 4. Assignment probability and transportation time data ($\omega_{l'\to l}|t_{l'(i)\to l(j)}$).

	Eval **(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**	**(8)**	**(9)**
***Retail*(1)**	1|1	1|2	1|3	1|4	1|5	1|6	1|7	1|8	1|9

	Rem **(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	***Distr*** **(1)**	**(2)**	**(3)**
***Eval*(1)**	0.67|1	0.67|2	0.67|3	0.67|4	0.67|5	0.67|6	0.33|1	0.33|2	0.33|3
***Eval*(2)**	0.67|1.5	0.67|2.5	0.67|3.5	0.67|4.5	0.67|5.5	0.67|6.5	0.33|1.5	0.33|2.5	0.33|3.5
***Eval*(3)**	0.67|2	0.67|3	0.67|4	0.67|5	0.67|6	0.67|7	0.33|2	0.33|3	0.33|4
***Eval*(4)**	0.67|2.5	0.67|3.5	0.67|4.5	0.67|5.5	0.67|6.5	0.67|7.5	0.33|2.5	0.33|3.5	0.33|4.5
***Eval*(5)**	0.67|3	0.67|4	0.67|5	0.67|6	0.67|7	0.67|8	0.33|3	0.33|4	0.33|5
***Eval*(6)**	0.67|3.5	0.67|4.5	0.67|5.5	0.67|6.5	0.67|7.5	0.67|8.5	0.33|3.5	0.33|4.5	0.33|5.5
***Eval*(7)**	0.67|4	0.67|5	0.67|6	0.67|7	0.67|8	0.67|9	0.33|4	0.33|5	0.33|6
***Eval*(8)**	0.67|4.5	0.67|5.5	0.67|6.5	0.67|7.5	0.67|8.5	0.67|9.5	0.33|4.5	0.33|5.5	0.33|6.5
***Eval*(9)**	0.67|5	0.67|6	0.67|7	0.67|8	0.67|9	0.670|10	0.33|5	0.33|6	0.33|7
***Mnfct*(1)**	0|0	0|0	0|0	0|0	0|0	0|0	1|10	1|20	1|30
***Retail*(1)**	1|1	1|1.5	1|2	1|2.5	1|3	1|3.5	1|10	1|20	1|30

Table 5. Computational performance.

	Generations	Mutations	Function Evaluations	Time (min)
Average	195662	357846	2663298	183
Standard Deviation	193239	311664	2683270	183

Table 6. Scenario performance.

		Input		***Output***							
Scenario		***SM***	***PM***	***SM***	***PM***	***REV***	***FC***	***VC***	***IC***	***Profit***	Δ
1	***DE***	3000	32100	1800	33000	6870000	0	2274000	164258	4431742	+1.8%
	QDS	3000	32100	1800	33000	6870000	0	2274000	241661	4354339	
2	***DE***	2993	32029	1796	32927	6854800	0	2268970	1130693	3455137	+23%
	QDS	3000	32100	1800	33000	6870000	0	2274000	1788072	2807928	
3	***DE***	2133	29669	1280	30309	6253800	0	2096030	511709	3646061	+0.2%
	QDS	2133	29669	1280	30309	6253800	0	2096030	517278	3640492	
4	***DE***	0	31237	0	31237	6247400	1852293	2186590	336110	1872407	+26.7%
	QDS	3000	32100	1800	33000	6870000	2954491	2274000	164258	1477251	

Table 7. Scenario 1.

		Eval								
		(1)	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**	**(8)**	**(9)**
DE	***Retail*(1)**	0	989	0	1013	0	0	998	0	0
QDS	***Retail*(1)**	333	333	333	333	333	333	333	333	333

		Rem						***Distr***		
		(1)	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(1)**	**(2)**	**(3)**
DE	***Eval*(2)**	1	2	0	298	292	0	1	295	1
	***Eval*(4)**	1	291	313	2	1	0	0	303	0
	***Eval*(7)**	295	0	1	1	0	302	298	1	0
	***Mnfct*(1)**							6398	6185	19517
	***Retail*(1)**	297	293	314	301	293	302	6698	6784	19518
QDS	***Eval*(*i*) $\forall i$**	33	33	33	33	33	33	33	33	33
	***Mnfct*(1)**							10700	10700	10700
	***Retail*(1)**	300	300	300	300	300	300	11000	11000	11000

Table 8. Scenario 2.

		Eval								
		(1)	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**	**(8)**	**(9)**
DE	***Retail*(1)**	1738	1255	0	0	0	0	0	0	0
QDS	***Retail*(1)**	600	533	467	400	333	267	200	133	67

		Rem						***Distr***		
		(1)	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(1)**	**(2)**	**(3)**
DE	***Eval*(1)**	795	0	208	40	0	0	0	226	296
	***Eval*(2)**	0	599	105	49	0	0	174	203	0
	***Mnfct*(1)**							31207	821	0
	***Retail*(1)**	795	599	312	89	0	0	31381	1250	296
QDS	***Eval*(1)**	103	86	69	51	34	17	90	60	30
	***Eval*(2)**	87	73	60	47	33	20	75	53	32
	***Eval*(3)**	73	62	52	41	31	21	62	47	31
	***Eval*(4)**	60	52	44	36	28	20	51	40	29
	***Eval*(5)**	48	42	36	30	24	18	42	33	25
	***Eval*(6)**	38	33	29	24	20	16	33	27	21
	***Eval*(7)**	28	25	22	18	15	12	24	20	16
	***Eval*(8)**	18	16	14	12	10	9	16	13	11
	***Eval*(9)**	9	8	7	6	5	4	8	7	6
	***Mnfct*(1)**							16050	10700	5350
	***Retail*(1)**	463	398	333	267	202	137	16450	11000	5550

Table 9. Scenario 3.

		Eval								
		(1)	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**	**(8)**	**(9)**
DE	***Retail*(1)**	238	239	234	238	239	235	238	238	234
QDS	***Retail*(1)**	237	237	237	237	237	237	237	237	237

		Rem						***Distr***		
		(1)	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(1)**	**(2)**	**(3)**
DE	***Eval*(1)**	0	0	0	0	142	0	70	1	0
	***Eval*(2)**	0	143	0	0	0	0	0	71	0
	***Eval*(3)**	0	0	0	0	72	68	0	70	0
	***Eval*(4)**	0	0	0	143	0	0	71	0	0
	***Eval*(5)**	0	0	0	0	0	143	0	72	0
	***Eval*(6)**	71	69	0	1	0	0	0	70	0
	***Eval*(7)**	0	0	143	0	0	0	0	0	71
	***Eval*(8)**	143	0	0	0	0	0	71	0	0
	***Eval*(9)**	0	0	71	70	0	0	0	0	70
	***Mnfct*(1)**							9888	9816	9965
	***Retail*(1)**	213	212	213	214	215	212	10101	10101	10107
QDS	***Eval*(*i*) ∀*i***	24	24	24	24	24	24	24	24	24
	***Mnfct*(1)**							9890	9890	9890
	***Retail*(1)**	213	213	213	213	213	213	10103	10103	10103

Table 10. Scenario 4: Fixed costs $\forall j$.

$\boldsymbol{Eval(j)}$	$\boldsymbol{Rem(j)}$	$\boldsymbol{Mnfct(j)}$	$\boldsymbol{Distr(j)}$
29342	14671	1389220	463073

Table A1. Overview of the notation for the network nodes.

$Mnfct$	Manufacturer
Rem	Remanufacturer
$Eval$	Evaluation centre
$Retail$	Retailer
$Distr$	Distribution
$Disp$	Disposal
PM	Primary market
SM	Secondary market

Table A2. Overview of the notation for the production process at network level l, with $l \in L = \{\text{Mnfct,Rem,Eval,Retail,Distr}\}$.

$WIP_{l(j)}$	average number of units at facility (j)
$EW_{l(j)}$	average lead time at facility (j)
$EWQ_{l(j)}$	average waiting time in the queue of batches at facility (j)
$Y_{l(j)(q)}$	binary decision variable to open facility (j) with capacity level q
$\lambda_{l(j)}$	average arrival rate at facility (j)
$\lambda_{l(j)(q)}$	average arrival rate at facility (j) with capacity level q
$C_{l(j)(q)}$	maximum production quantity of facility (j) with capacity level q
$t_{l(j)}$	average 'effective' process time in facility (j)
$\mu_{l(j)}$	average 'effective' process rate in facility (j)
$\rho_{l(j)}$	utilisation of facility (j)
$[c^2_{l(j)}]_p$	SCV of the unit production time in facility (j)
$[c^2_{l(j)}]_{bp}$	SCV of the batch production time in facility (j)
$[c^2_{l(j)}]_{bd}$	SCV of the batch interdeparture time from facility (j)
$[c^2_{l(j)}]_{ba}$	SCV of the batch interarrival time at facility (j)
$FIX_{l(j)(q)}$	fixed cost for facility (j) with capacity level q
$vc_{l(j)}$	variable unit cost in facility (j)
$dc_{l(j)}$	unit disposal cost in facility (j)
$hc_{l(j)}$	unit holding cost in facility (j)

Table A3. Overview of the notation for the transportation process between facility (i) at network level l' and facility (j) at network level l, with $l, l' \in L = \{Mnfct, Rem, Eval, Retail, Distr\}$ and $l' \neq l$.

$WIP_{l'(i)\to l(j)}$	average number of units
$EW_{l'(i)\to l(j)}$	average lead time
$EWQ_{l'(i)\to l(j)}$	average batch waiting time in the queue
$WTBT_{l'(i)\to l(j)}$	average wait-to-batch-time
$\lambda_{l'(i)\to l(j)}$	average arrival rate $(l, l' \in L \cup \{PM, SM\})$
$[\lambda_{l'(i)\to l(j)}]_{ba}$	average batch arrival rate
$[c^2_{l'(i)\to l(j)}]_{ba}$	SCV of the batch interarrival times
$[c^2_{l'(i)\to l(j)}]_{bd}$	SCV of the batch interdeparture times
$t_{l'(i)\to l(j)}$	average 'effective' transportation time
$[c^2_{l'(i)\to l(j)}]_{t}$	SCV of the 'effective' transportation time
$\rho_{l'(i)\to l(j)}$	transportation utilisation
$m_{l'(i)\to l(j)}$	number of transportation vehicles
$\omega_{l'\to l}$	predefined fraction to direct the flow between two network levels, used for flow feasibility, grade classification and scrap $(l \in L \cup \{Disp\})$
$\varsigma_{l'(i)\to l(j)}$	undefined fraction to direct the flow between (i) and (j), a decision variable with $l, l' \in L \cup \{PM, SM\}$)
$d_{l'(i)\to l(j)}$	distance
$tc_{l'(i)\to l(j)}$	distance unit cost
$hc_{l'(i)\to l(j)}$	unit holding cost
B	uniform batch size

Table A4. Overview of the notation for the supply/demand process.

$R_{PM(1)}$	supply of returns
$R_{RM(1)}$	supply of raw materials
$D_{PM(1)}$	demand for new products
$D_{SM(1)}$	demand for remanufactured products
$[c^2_{PM(1)\rightarrow Retail(j)}]_a$	SCV of unit interarrival times at retailer (j)
$[c^2_{RM(1)\rightarrow Mnfct(j)}]_a$	SCV of unit interarrival times at manufacturer (j)
$pcr_{PM(1)}$	unit penalty cost for uncollected returns from PM
$pcd_{PM(1)}$	unit penalty cost for unsatisfied demand in PM
$pcd_{SM(1)}$	unit penalty cost for unsatisfied demand in SM
$pr_{PM(1)}$	unit selling price in PM
$pr_{SM(1)}$	unit selling price in SM

Figure 1: Structure of the CLSC.