Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computers & Operations Research 34 (2007) 395–416
computers & operations research
www.elsevier.com/locate/cor

Reverse logistics network design with stochastic lead times

Kris Lieckens*, Nico Vandaele

Faculty of Applied Economics, University of Antwerp, Prinsstraat 13-2000 Antwerp, Belgium

Available online 27 April 2005

Abstract

This work is concerned with the efficient design of a reverse logistics network using an extended version of models currently found in the literature. Those traditional, basic models are formulated as mixed integer linear programs (*MILP*-model) and determine which facilities to open that minimize the investment, processing, transportation, disposal and penalty costs while supply, demand and capacity constraints are satisfied. However, we show that they can be improved when they are combined with a queueing model because it enables to account for (1) some dynamic aspects like lead time and inventory positions, and (2) the higher degree of uncertainty inherent to reverse logistics. Since this extension introduces nonlinear relationships, the problem is defined as a mixed integer nonlinear program (*MINLP*-model). Due to this additional complexity, the *MINLP*-model is presented for a single product-single-level network. Several examples are solved with a genetic algorithm based on the technique of differential evolution.

Keywords: Location; Queueing; Network flows; Supply chain management; Differential evolution

1. Introduction

The efficient design of a product recovery network is one of the challenging issues in the recently emerged field of reverse logistics. Especially large-scale businesses are concerned with the optimization of the location and the capacity of their facilities, as well as the product flows between them. Due to the high number of processing activities involved and the high variability in the value of the components, this location-allocation problem is particularly relevant in a reverse logistics context.

However, the high level of uncertainty in this field also applies to network design. In particular, compared to the traditional forward chain, it is more difficult to control for the timing, quantity and quality

* Corresponding author. Tel.: +32 3 220 40 17; fax: +32 3 220 47 99.
E-mail address: kris.lieckens@ua.ac.be (K. Lieckens).

0305-0548/$ - see front matter
doi:10.1016/j.cor.2005.03.006

of the supply, while in addition the processing activities are highly variable [1]. Furthermore, current deterministic facility location models lack the ability to incorporate inventory related costs, as well as a product's lead time through the network. Nevertheless, the time that returned products spend in the system is an important issue because the costs of obsolescence may be significant.

This paper attempts to overcome these shortcomings by extending current mixed integer linear program models (*MILP*) with some queueing characteristics using a $G/G/m$ model. However, the application of this methodology to integer location problems is challenging. These models are hard to solve by themselves, so combining them increases the computational complexity even more. Therefore, we restrict ourselves to a single-level, single-product network design problem that covers a single period. The introduction of queueing nonlinearities in the objective function transforms the traditional model into a mixed integer nonlinear program model (*MINLP*) and forces us to apply other solution methods. With reference to the successful use of the differential evolution technique (*DE*) in a large number of difficult design and control *MINLP* problems in the chemical engineering [2], we opt to apply this algorithm to our supply chain-related problem.

The use of queueing theory for network design including variabilities can also be found in models for the citing of emergency vehicles under congestion and randomness. Marianov and ReVelle [3] maximized the demand covered with a given reliability by determining how to cite limited numbers of emergency vehicles. They used an $M/G/m$ model and applied linear programming relaxation with Brach and Bound to solve the problem. Marianov and Serra [4] also developed several probabilistic maximal covering location-allocation models with constraints on the number of elements in the queue. Based on the $M/M/m$ model, they used linear equivalent formulations in combination with Branch and Bound and heuristics methods to solve it. Finally, Amiri [5] includes a waiting time cost in the objective function for the design of service systems, also using an $M/M/m$ model. Two heuristics are applied based on a Lagrarian relaxation and decomposition of the problem. Nevertheless, these solution techniques cannot be applied when a $G/G/m$ model is used.

The structure of this work is as follows. We first describe the modeling problem in Section 2 and then in Section 3, we present the new *MINLP*-model, which enables us to locate recovery facilities and assign returned goods in a cost efficient way including inventories, lead times and variabilities. Section 4 deals with the technique of differential evolution as the solution method for this new model. Some examples will illustrate the application of the algorithm. Finally, in Section 5, we draw conclusions and outline future research.

2. Problem definition

Used products of a single type return in various quantities and with some variability to a finite number of given collection locations. This flow is the supply of returned products into the reverse logistic system. For simplicity, suppose that no activity is performed at this location and that this flow is immediately transported to a facility where only one particular process is carried out (e.g. disassembly, recycling, remanufacturing, etc.). Reprocessing will reveal that not all arriving parts meet the production and quality norm, which results in some fraction that is disposed of. Since this study is limited to a single-level network, we assume that finished goods are immediately sold (i.e. without storing and transporting them) to a finite number of given customers, which generates some revenue (*REV*). Inventories do occur at the facility locations. We also assume that the volume of supply and demand, as well as the fractions

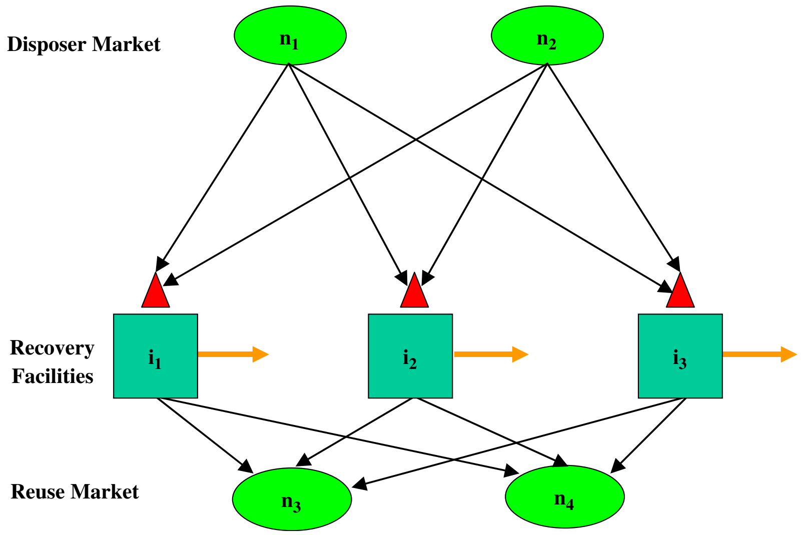

Fig. 1. Structure of a reverse logistics network.

to be disposed of can be estimated based on historical and technical data and/or future expectations. Fig. 1 is a visual representation of this network, where the disposer market or collection stations serve as a source, the recovery facilities serve as intermediary nodes and the reuse market serves as a sink. This graph incorporates all possible facilities and transportation links.

We recommend the set of candidate locations is specified in advance. In this way, we both limit the search space and ensure that the final decision also satisfies other less tangible strategic variables. Next, all facilities can be installed to operate at a set of predetermined capacity levels, each at different cost. The fixed cost for reprocessing and capital investment ($=FC$) rises with the capacity level, whereas the variable unit cost for processing ($=VC$) declines due to economies of scale. In addition, an inventory cost (IC) associated with the inventory levels may become significant, especially in case of high valuable goods (i.e. electronic equipment). The longer a product stays in the network, the more it devaluates. Therefore, this cost of obsolescence is also included in the IC, which is expressed as a cost per unit per year. Furthermore, the transportation cost ($=TC$) is assumed to be a linear function of volume.

In order to balance supply and demand and to satisfy downstream capacity constraints, the disposal option is used as a sink for excess supply. To this end, a minimum disposal fraction is introduced that is required for products that cannot be reused for technical or economic reasons. If the supply of return goods exceeds its demand, this surplus will also be discarded and the disposal fraction will rise above its minimum. One can argue that this surplus should not be transported, but we assume that the quality can only be assessed after recovery operations; so discarding the surplus at the collection stations would involve the danger of throwing away high-quality components. A cost is evidently associated with the total disposed quantity ($=DC$). Another option to balance supply and demand is not collecting some fraction of the returned products or not satisfying some fraction of demand, which both involve penalty costs ($=PC$). The uncollected fraction is not allowed to enter the network.

To conclude, the recovery network design problem can be summarized and defined as follows: "given the locations of customers, the potential locations for reprocessing facilities and their capacity levels, the volume of both supply and demand for one product type, the variabilities in returns and reprocessing, the cost structure (fixed, variable, transportation, inventory, disposal and penalty costs), we try to determine which facilities should be operated at which capacity level and how the flow should be assigned so that the overall expected yearly profit of the system is maximized". This stochastic single-period, single-echelon and single-product facility-location-allocation problem that can handle the variabilities in the network will be formulated as a *MINLP* model in the next section.

3. The reverse logistics network MINLP-model

3.1. Motivation

The model for the recovery network design presented below is a single-product single-level location model with a time horizon of 1 year that is based on the models of Krikke [6] and Fleischmann [7]. As *MILP* is an established solution approach for these traditional facility allocation models, they used the same approach for the recovery network design problem with some adjustments to reflect their specific characteristics. Binary variables are used to decide whether a facility is open (=1) or not (=0), while continuous variables indicate the assigned flow of the goods. However, this *MILP*-model does not take into account the cycle time, which is the time spent by the products in the system, while nowadays, firms who are able to deliver finished products fast clearly have a competitive advantage. Therefore, we extend this model with the cycle time by introducing queueing relationships into the network.

More fundamentally, a queueing system can also deal with the variability of the arrival process and processing steps. A squared coefficient of variation (*SCV*), which is equal to the variance of the time relative to the square of its mean, is used to gain insight in the variability of these processes. This is of great relevance in a reverse logistics context because these networks are characterized by a high degree of uncertainty caused by the lack of control over the quantity, quality and timing of returned products [1]. As a result of different use patterns by customers, the quality of returned parts is highly variable, causing process variabilities that can raise the process *SCV* to higher levels compared to those observed in traditional forward chains.

In addition, the expected lead times can be transformed into expected queue lengths based on Little's law ($L=\lambda W$). As a result, we can integrate the queueing relationships into the *MILP*-model by extending the objective function with inventory holding costs, transforming it into a *MINLP*-model. Since the network design is a long-term strategic decision, we can argue that steady-state conditions are reached. This section describes this new contribution in more detail. We refer to Fig. 2 for an overview of the notation.

3.2. Parameters and performance measures

The following parameters are assumed to be known:

Index sets:

$I=\{1,\ldots,i,\ldots,I_{\max}\}$	set of reprocessing locations i,
$N=\{1,\ldots,n,\ldots,N_{\max}\}$	set of customer locations n (both disposer and reuse markets or $N^{\mathrm{Disp}}\subset N$ and $N^{\mathrm{Reuse}}\subset N$),
$Q=\{1,\ldots,q,\ldots,Q_{\max}\}$	set of capacity levels q (discrete values).

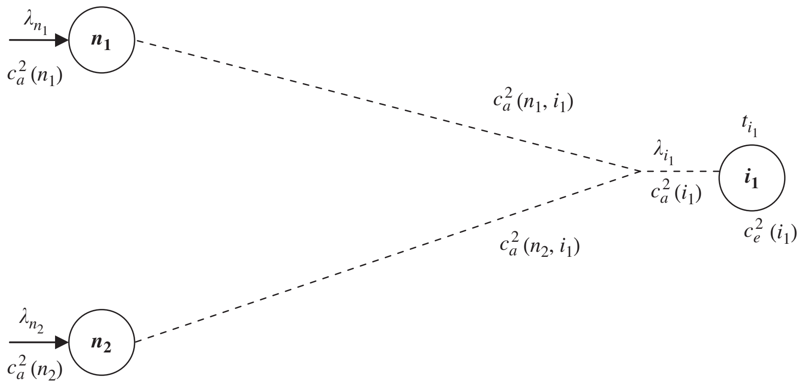

Fig. 2. Notation overview.

Decision variables:

X_i total product flow, not including disposed fractions, reprocessed at facility i,
$X_i(q)$ total product flow reprocessed at facility i, installed at capacity q,
X_{n_i} product flow from disposer market n to facility location i,
X_{i_n} product flow from facility location i to reuse market n,
$X_{i_{\text{disp}}}$ product flow disposed of at facility i,
$Y_i(q)$ boolean indicating whether facility i is installed at capacity level q (value $= 1$) or not (value $= 0$),
u_n fraction of demand not satisfied for reuse customer n,
w_n fraction of returns not collected from disposer customer n.

Costs and revenue parameters:

$FIX_i(q)$ yearly fixed cost to open facility i, installed at level q,
$c_i(q)$ unit reprocessing cost of product flow at facility i operating at level q,
$c_i(h)$ unit holding cost per year at facility i,
c_{n_i} unit transportation cost between disposer market n and facility location i,
c_n^u unit penalty cost for not satisfying demand of reuse customer n,
c_n^w unit penalty cost for not collecting returns from disposer customer n,
$c_{i_{\text{disp}}}$ unit disposal cost at facility i,
p_n unit selling price to reuse customer n.

Queueing parameters:

t_i mean effective reprocessing time at facility i,
$c_{\mathrm{a}}^2(n)$ *SCV* of the collected arrivals at disposer market n,
$c_{\mathrm{e}}^2(i)$ *SCV* of effective reprocessing time at facility i.

Other parameters:

D_n yearly demand of reuse customer n,
R_n yearly returns from disposer customer n,
$C_i(q)$ maximum capacity of facility i when installed at capacity level q,
γ minimum disposal fraction.

Generally, a queueing system is characterized by an arrival process, a process step and a queue [8]. The arrivals at the different nodes in our network are equivalent to the flow variables of the *MILP*-model, while each facility at the first level in the network graph in Fig. 1 is the processing step. As a result, the expected number of arrivals at each facility i per time unit, which are expressed by the arrival rate λ_i, are the decision variables. Their inverse value represents the mean interarrival time. The mean effective reprocessing rate μ_i, which is the inverse of t_i, is equivalent to the maximum capacity level $C_i(q)$ at which facility i is installed:

$$t_i = \frac{1}{\mu_i} = \frac{1}{\sum_q C_i(q)Y_i(q)}. \tag{1}$$

We are interested in the following performance measures for each facility i:

$E(WQ_i)$	expected waiting time in the queue at facility i,
$E(W_i)$	expected time spent at facility i (i.e. queue time plus reprocessing time),
$E(NQ_i)$	expected number of products in the queue at facility i,
$E(N_i)$	expected number of products at facility i.

3.3. Mathematical model

The recovery network design model is now formulated as follows:
Max

$$\sum_i \sum_n p_n X_{i_n} \tag{REV}$$

$$-\Bigg(\sum_q \sum_i FIX_i(q)Y_i(q) \tag{FC}$$

$$+\sum_q \sum_i c_i(q)X_i(q) \tag{VC}$$

$$+\sum_i c_i(h)E(N_i) \tag{IC}$$

$$+\sum_n \sum_i c_{n_i}X_{n_i} \tag{TC}$$

$$+\sum_n c_n^u D_n u_n + \sum_n c_n^w R_n w_n \tag{PC}$$

$$+\sum_i c_{i_{\text{disp}}} X_{i_{\text{disp}}}\Bigg) \tag{DC}$$

subject to (2)

Balance constraints

Reprocessing facility:

$$R_n(1-w_n)=\sum_i X_{n_i} \quad \forall n, \tag{3}$$

$$\sum_n X_{n_i}=\sum_q X_i(q) \quad \forall i, \tag{4}$$

$$\gamma \sum_q X_i(q) \leqslant X_{i_{\text{disp}}} \quad \forall i, \tag{5}$$

$$\sum_q X_i(q)-X_{i_{\text{disp}}}=X_i \quad \forall i, \tag{6}$$

$$X_i=\sum_n X_{i_n} \quad \forall i. \tag{7}$$

Demand:

$$\sum_i X_{i_n}=D_n(1-u_n) \quad \forall n. \tag{8}$$

Capacity constraints

$$X_i(q) \leqslant C_i(q)Y_i(q) \quad \forall i \quad \forall q, \tag{9}$$

$$X_i(q) \geqslant C_i(q-1)Y_i(q) \quad \forall i \quad \forall q, \tag{10}$$

$$\sum_q Y_i(q) \leqslant 1 \quad \forall i. \tag{11}$$

Logical constraints

$$\begin{aligned} Y_i(q)=\{0,1\} &\quad \forall i \quad \forall q, \\ X_i(q) \geqslant 0 &\quad \forall i \quad \forall q, \\ X_{n_i} \geqslant 0,\ X_{i_n} \geqslant 0,\ X_i \geqslant 0 &\quad \forall n \quad \forall i, \\ 0 \leqslant w_n \leqslant 1 &\quad \forall n, \\ 0 \leqslant u_n \leqslant 1 &\quad \forall n. \end{aligned}$$

The set N of customer locations n includes both disposer and reuse markets in order to facilitate the modeling of both open- and closed-loop supply chains depending on the customer's demand and return quantities. In a closed-loop supply chain, those who reuse the products are the same as those who have disposed them. This means that a particular customer n belongs to a closed loop when his demand and return quantities are both positive. On the other hand, there is a distinction between both markets in an open loop, which is true when either the demand or the return quantity of a particular customer n is zero.

The balance constraints that link up the input and output streams at a node or facility need some explanation. Constraint (3) ensures that all, or at least a part of the returned products leave the collection stations at each location n. All the incoming flow at each facility i will be reprocessed in constraint (4), after which the minimum fraction is disposed of in constraint (5). The amount of useful products at facility i after reprocessing equals the total incoming flow minus the disposed quantity (see constraint (6)) and is send to the reuse market n in constraint (7). Constraint (8) reflects the requirement that product's demand at the reuse market n must be satisfied, otherwise some penalty cost will be added to the objective function.

Since it is difficult to determine the cost of the cycle time of a particular product through the system, it is not included in the objective function. Fortunately, by Little's law ($E(N_i) = \lambda_i E(W_i)$), we can implicitly take into account the cycle time cost through the holding costs. It is the cost of holding one unit in stock for one year because the objective function is defined on a yearly basis as well. An expression for this expected yearly inventory cost is derived next. Note that without the *IC*, this model is transformed into the traditional *MILP*-model.

3.4. The queueing relationships

3.4.1. General issues

A $G/G/1$ queueing model is used with general interarrival and process time distributions, infinite buffers and a facility modeled as a single server. Unfortunately, the performance measures for the $G/G/1$ queue cannot be determined in an exact analytic way. We rely on a two-moment approximation, which is based on the mean and the variance of the interarrival and process time distributions.

The approximation for the expected waiting time (*EW*) in the $G/G/1$ queue, which was first developed by Kingman [8], is given by

$$EW(G/G/1) = \left(\frac{c_\mathrm{a}^2 + c_\mathrm{e}^2}{2}\right)\left(\frac{\rho}{1-\rho}\right)\frac{1}{\mu} = \left(\frac{c_\mathrm{a}^2 + c_\mathrm{e}^2}{2}\right) EW(M/M/1), \tag{12}$$

where $EW(M/M/1)$ is the expected waiting time with exponential arrival and exponential service distributions, and a single server. However, we prefer the approximation from Whitt because it performs fairly well in most circumstances due to a correction factor ϕ [9]:

$$EW(G/G/1) = \phi\left(\rho, c_\mathrm{a}^2, c_\mathrm{e}^2, 1\right)\left(\frac{c_\mathrm{a}^2 + c_\mathrm{e}^2}{2}\right)\left(\frac{\rho}{1-\rho}\right)\frac{1}{\mu}. \tag{13}$$

We refer to Appendix C for more details about this correction factor. The utilization level ρ in (12) and (13) is equal to λ/μ and must be strictly smaller than 100% to avoid system explosion. Although there exists exact values for the $M/M/1$ model, only the heavy traffic approximation is shown in these equations [9]. They reveal that the expected waiting time increases in a nonlinear way when the utilization level is increased.

In order to develop a network model, we rely on a decomposition method to obtain approximate results for the performance of such systems. This methodology consists of three consecutive steps [10]. Firstly, in order to analyze the interaction between stations of the network, the network is regarded as a composite of three basic processes [11], which are addressed in Section 3.4.2.

1. Aggregation.
2. Flow through a queue or station.
3. Decomposition or splitting.

Secondly, the network will be decomposed into subsystems of individual stations. The performance of each subsystem can now be analyzed using the information from step 1 and the $G/G/1$ approximation

(13). In step 3, the network performance can be analyzed by synthesizing the results of all subsystems in step 2.

3.4.2. Reprocessing facility

The disposer market N is the main source of our open queueing network. At the transportation line between a particular collection point $n \in N^{\mathrm{Disp}}$ and a reprocessing facility $i \in I$, individual products arrive with a rate λ_{n_i} equal to X_{n_i}, which is just a part of the total return arrival rate λ_n at collection location n with

$$\lambda_n = R_n(1 - w_n) = \sum_i X_{n_i}. \tag{14}$$

At facility i, the individual arrival rate equals

$$\lambda_i = \sum_n \lambda_{n_i} = \sum_q X_i(q). \tag{15}$$

Using the relationship between the expected number of elements $E(N_i)$, the expected cycle time $E(W_i)$ and the arrival rate λ_i according to Little's law, we can formulate the expected yearly inventory costs at facility i as

$$c_i(h)[E(N_i)] = c_i(h)[\lambda_i E(W_i)]. \tag{16}$$

The expected cycle time of a product $E(W_i)$, consists of the following components:

- Expected waiting time of individual elements in the queue $E(WQ_i)$.
- Expected reprocessing time t_i.

So we have

$$E(W_i) = E(WQ_i) + t_i. \tag{17}$$

An expression for the $E(WQ_i)$ is formulated now. Based on the information about the *SCV* of the individual arrivals at the collection location n $\left(c_{\mathrm{a}}^2(n)\right)$, we can derive an expression for the *SCV* of the individual arrivals at facility i $\left(c_{\mathrm{a}}^2(i)\right)$. We refer to Appendix B for more detail. Next, the effective utilization level ρ_i must be established where the effective reprocessing time is equal to Eq. (1)

$$\rho_i = \lambda_i t_i \leqslant 1. \tag{18}$$

Note that the *MILP*-model already guaranteed system stability (constraint (18) is similar to the set of constraints (9)–(11)) and that a facility is modeled as a single server ($m = 1$). Now all the needed components ((1), (18) and $c_{\mathrm{a}}^2(i)$) are known to formulate the expected waiting time of a product at facility i according to Eq. (13)

$$E(WQ_i) = \phi\left(\rho_i, c_{\mathrm{a}}^2(i), c_{\mathrm{e}}^2(i), 1\right)\left(\frac{c_{\mathrm{a}}^2(i) + c_{\mathrm{e}}^2(i)}{2}\right)\frac{\rho_i}{(1-\rho_i)}\, t_i. \tag{19}$$

We conclude this section with the overall expression for the inventory cost component IC in Eq. (2), which is found by combining Eqs. (14)–(19)

$$IC = \sum_i c_i(h) \left(\sum_q X_i(q) \right) \frac{1}{\sum_q C_i(q) Y_i(q)} \times \left[\phi \left(\frac{c_a^2(i) + c_e^2(i)}{2} \right) \frac{\sum_q X_i(q)}{\sum_q C_i(q) Y_i(q) - \sum_q X_i(q)} + 1 \right]$$

with $c_a^2(i)$ equal to Eq. (21) in Appendix B.

4. Computational results

4.1. Solution methods

Since the expected cycle time increases in a nonlinear way when the utilization level is increased, the same relationship also holds for the expected number of elements in the system, and the expected inventory costs. The idea is that congestion may lead to an infinite inventory position and that uncertainty makes this even worse. Consequently, the type of optimization decision required for the new model can be classified under the mixed integer nonlinear programming problems. Due to their combinatorial nature, the presence of continuous and binary variables considerably increases the difficulty of finding the global optimal solution, certainly in a nonlinear search space.

This nonlinear relationship will affect profit as displayed in Fig. 3, which plots the profit on the z-axis, for various levels of the flow assigned to two potential facility locations, indicated respectively on the x- and y-axis. In case of more realistic networks, the form of this function is not obvious with the

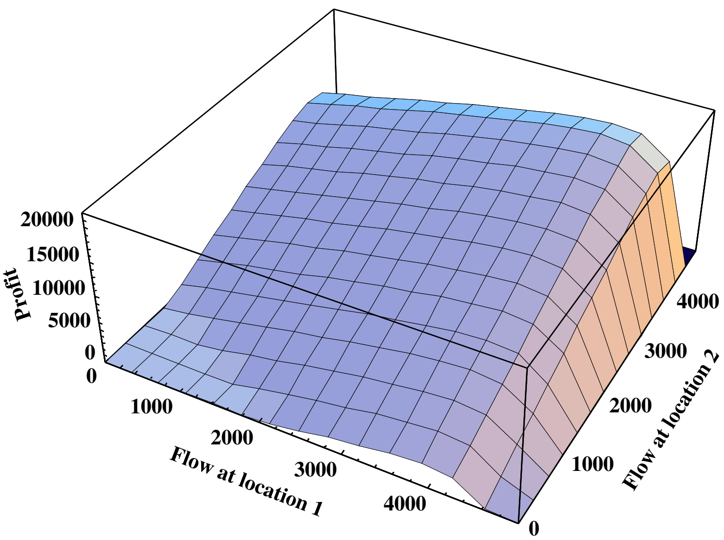

Fig. 3. Nonlinear profit relationship for two potential locations.

possibility of local optima, and compared to the *MILP* solution another flow distribution or even another layout may become more profitable. Investigation of the form in Fig. 3 for different parameter settings revealed that the higher the level of all the linear cost components relative to the nonlinear inventory cost component, the lower the impact of the latter will be on the final decision. This suggests that the *MINLP*-model is only relevant when costs related to inventory, obsolescence and the product's lead time (i.e. customer satisfaction) are significant.

The methods for solving *MINLP* problems are usually divided into two main categories: deterministic and stochastic. Deterministic methods, such as branch-and-reduce [12,13] the α BB branch-and-bound [14,15], Bartson's branch-and-cut [16], interval analysis-based methods [17] and the extended cutting plane [18], run through the algorithm in a predefined way, which means that the next step of computation is exactly determined. On the other hand, stochastic global optimization methods (also known as adaptive random search methods), such as differential evolution [19], adaptive Lagrange-multiplier methods [20], simulated and nested annealing [21], ant colony simulation [22] and tabu search [23], run through the algorithm in a random way, which means that the next step of computation is undetermined.

4.2. Differential evolution

In recent years, stochastic algorithms have solved *MINLP* problems in many engineering applications, mostly in the area of chemical engineering [24]. Babu and Angira [2] found the technique of *DE* to be the best evolutionary computation method after the study of seven difficult design and control *MINLP* problems in chemical engineering. *DE* proved to be simple, fast and robust. It is basically a computerized search and optimization algorithm and is likely to find a function's true global optimum. In addition, it can easily be implemented in a parallel computing environment, which speeds up the optimization. Therefore we opt to use *DE* to solve our network design problem.

DE is an improved version of the genetic algorithm, which belongs to the class of evolutionary algorithms that are based on the principle of survival of the fittest. 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 $\overrightarrow{X}_{p,G}$. Each vector $\overrightarrow{X}_{p,G}$ consists of D decision variables $X_{o,p,G}$. Each flow variable X_{n_i} ($\forall n \in N^{\mathrm{Disp}}, \ \forall i \in I$) and X_{i_n} ($\forall i \in I, \ \forall n \in N^{\mathrm{Reuse}}$) as well as each binary variable $Y_i(q)$ ($\forall i \in I, \forall q \in Q$) is considered as such a decision variable. As a result, D equals $N^{\mathrm{Disp}} * I * Q + I * N^{\mathrm{Reuse}} + I * Q$. This is briefly summarized as [25]:

$$P_G = \left\{ \overrightarrow{X}_{1,G}, \overrightarrow{X}_{2,G}, \ldots, \overrightarrow{X}_{p,G}, \ldots, \overrightarrow{X}_{NP,G} \right\},$$

$$\overrightarrow{X}_{p,G} = \{X_{1,p,G}, X_{2,p,G}, \ldots, X_{o,p,G}, \ldots, X_{D,p,G}\},$$

$$p = 1, 2, \ldots, NP, \quad o = 1, 2, \ldots, D, \quad NP \geqslant 4, \quad G = 1, \ldots, G_{\max}.$$

The detailed *DE* algorithm used here is given below.

- *Choose a strategy*: Price and Storn suggested ten different strategies of *DE*. We will choose from the *DE/rand/1/bin* and the *DE/current-to-rand/1* schemes, which are explained in the next steps.
- *Initialize the key parameters of control*: The crossover constant *CR*, the population size *NP*, the mutation scaling factor *F*, the coefficient of combination *K* and the maximum number of generations $G_{\max}$, are the user-defined control parameters. Again, we refer to the next steps for more information.

Their values remain constant during the search process. The two control parameters F and K appear to be quite closely related. Although K is continuous, it is mostly sufficient to set it to one of the three discrete values: 0, 0.5, 1. Generally, F, K and CR affect the convergence velocity and robustness of the search process. Their optimal values depend both on problem characteristics and on NP, and can be found by trial and error. Based on their extensive experience, Price and Storn [19] give guidelines to select values for these control parameters. In order to prevent premature convergence or stagnation, a CR-value close to 1, $NP = 20D$, $K = 0.5$ and $F = 0.8$ appear to be adequate starting points in real-world design problems. In case of premature convergence, NP and/or F should be increased, and/or K decreased. In case of stagnation of the population, NP and/or F should be increased, and/or K randomly chosen from the interval [0,1]. In addition, they suggest a population size NP up to $100D$ for functions with many of local optima.

- *Initialize the population*: Initialization of the population provides us with a starting solution for optimum seeking. The initial population $P_{G=0}$ is chosen randomly and should cover the entire variable space. Note that for each decision variable o, another random value is generated.

$$X_{o,p,G=0} = X_o^{(lo)} + rand_o[0,1]\left(X_o^{(hi)} - X_o^{(lo)}\right), \quad p = 1, 2, \ldots, NP, \; o = 1, 2, \ldots, D.$$

  The term $rand_o$ represents a uniformly distributed random variable that ranges from zero to one, whereas the superscripts *hi* and *lo* denote upper and lower initial parameter bounds, respectively.

In order to maintain the diversity of the population and the robustness of the algorithm, DE always works with a population of continuous variables regardless of the corresponding decision variable type (i.e. continuous or binary). Nevertheless, binary values for the open or close variables $Y_i(q)$ $(\forall i, \forall q)$ in the trial vector should be used to evaluate the objective function. These conditions are accomplished by first initializing the binary variables as

$$\begin{aligned} X_{o,p,G=0} &= X_o^{(lo)} + rand_o[0,1]\left(X_o^{(hi)} - X_o^{(lo)} + 1\right) \\ &= rand_o[0,1](2), \quad p = 1, 2, \ldots, NP, \; o = 1, 2, \ldots, D \end{aligned}$$

followed by truncation for evaluation, while DE itself still keeps on working internally with the continuous values [26].

- Evaluate the profit of each vector.
- Find out the vector in the population with the highest profit.
- Perform mutation and recombination.

'Mutation' aims to keep a population robust and to search a new area. Mutation involves adding a randomly generated step to one or more parameters of an existing object vector in order to move existing object vectors in the right direction by the right amount at the right time. DE mutates an object vector by adding the weighted difference of a randomly sampled pair of vectors in the current population P_G to a third one. F is the weight applied to this random differential variation and scales the magnitude of the step size. Effective values for F belongs to the interval (0, 1+]; larger values can be used if needed, but do not appear to be productive [26]. In case of *DE/rand/1/bin*, the population of child or trial vectors

$P_{G+1} = \overrightarrow{U}_{p,G+1} = U_{o,p,G+1}$ for each parent or target vector $X_{o,p,G}$ (where $p=1,\ldots,NP, o=1,\ldots,D$) is created as follows:

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

where

$k \in \{1,\ldots,D\}$, random decision variable index,

$r_1, r_2, r_3 \in \{1, 2, \ldots, NP\}$, randomly selected, but $r_1 \neq r_2 \neq r_3 \neq p$,

$CR \in [0, 1]$, $F \in (0, 1+]$, and $R(o) \in [0, 1]$ is a uniformly random number.

The notation */rand/* means that the donors to be mutated are randomly chosen from the population members. Also, the randomly chosen indices r_1, r_2 and r_3 must be mutually different and different from *p*, which stands for the current parent object vector. Consequently, *NP* must be greater than 3. New, random, integer values for r_1, r_2 and r_3 are chosen for each value of *p*, i.e. for each individual candidate solution. 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.

'Recombination', or crossover, is complementary to mutation and reinforces prior successes by building trial vectors out of existing object vector parameters. The crossover operation creates a trial vector $\overrightarrow{U}_{p,G+1}$ by selecting elements from the target vector $\overrightarrow{X}_{p,G}$ and the randomly chosen, mutated donor vector $\overrightarrow{V}_{p,G+1}$. The binomial scheme ($/bin/$) takes parameters from $\overrightarrow{V}_{p,G+1}$ every time that $R(o) \leqslant CR$, otherwise the parameters come from $\overrightarrow{X}_{p,G}$. In other words, the *CR* value controls the probability that a trial vector parameter will come from the randomly chosen, mutated vector $\overrightarrow{V}_{p,G+1}$, instead of from the current vector $\overrightarrow{X}_{p,G}$. In the *DE/current-to-rand/1* scheme, *CR* does not need to be specified because it is implicitly equal to 1. Child vectors are here created as

$$\overrightarrow{U}_{p,G+1} = \overrightarrow{X}_{p,G} + K\left(\overrightarrow{X}_{r_3,G} - \overrightarrow{X}_{p,G}]\right) + F\left(\overrightarrow{X}_{r_1,G} - \overrightarrow{X}_{r_2,G}\right),$$

where

$r_1, r_2, r_3 \in \{1, 2, \ldots, NP\}$, randomly selected, but $r_1 \neq r_2 \neq r_3 \neq p$,

$K \in [-0.5, 1.5]$.

- *Check lower and upper bounds of the variables*: The parameters of the child vectors must be checked for boundary conditions. If some parameter violates its corresponding boundary condition, then it is randomly assigned a value in the bound range.
- *Perform selection*: After each child vector is evaluated by the objective function, its profit is compared to the profit of its parent. The vector with higher profit is selected for the next generation. As a result, all the individuals of the next generation are good as or better than their counterparts in the current generation.
- *Repeat the evolutionary cycle until $G_{\max}$ is reached*: Constraint Eqs. (3)–(8), (11) and (18) are implemented as 'soft' constraints. This means that penalty functions cause an object vector's profit to

Table 1
Different network configurations

	Case 1	Case 2
Collection locations	$n1, n2$	$n1, n2$
Facility locations	$i1, i2, i3$	$i1, i2, i3$
Capacity levels	$q2$	$q1, q2, q3$
Demand locations	$n3, n4$	$n3, n4$

decrease with both the magnitude and number of its constraint violations. The objective function is reformulated as follows:

$$Profit(X, Y) = REV(X)\left(FC(Y) + VC(X) + TC(X) + PC(X) + DC(X) + IC(X) + \sum_{m=1}^{v} c_m(X, Y)\right),$$

where

$$v = \text{number of constraints},$$
$$c_m(X, Y) = MAX[s_m g_m(X, Y), 0] \quad \forall m \in \{1, 2, \ldots, v\},$$
$$s_m \geqslant 1 \quad \forall m \in \{1, 2, \ldots, v\}.$$

Each constraint is reformulated in such a way that it is greater than zero when it is violated. In general, this new constraint form is represented by $g_m(X, Y)$. In combination with a high value of the constant s_m, a constraint violation will now decrease the profit to such an extent that this situation will never be considered as the optimum.

4.3. Examples

Two different network design problems with increasing complexity (see Table 1) are investigated for the input parameters found in Appendix A. For each case, the *MILP*-model is solved with *LINDO*, while the *MINLP*-model is solved with the *DE*-algorithm described in Section 4.2. The results are summarized in Tables 3 and 4. All computations are performed on a Pentium 3, 732 MHz.

Three independent runs, each with a different *DE* setting, are applied to each case to investigate the convergence rate and the robustness of the *DE* solution. The choice of the *DE* schemes and their parameter setting were based on the guidelines of Price and Storn [19] and can be found in Table 2. Setting 1 is the *DE/current-to-rand/1* scheme with $F = 0.8$ and $K = 0.5$. In setting 2, the effect of a slower convergence rate for this scheme is investigated by increasing the value for F and decreasing the value for K ($F = 0.9$ and $K = 0.2$). Setting 3, the *DE/rand/1/bin* scheme with $F = 0.9$ and $Cr = 0.8$, is expected to be much slower. In order to compare computation time, *NP* is set to $50D$ in all the three settings.

To have an idea about the performance quality of the used heuristic, we simply compared its outcome to the best solution found by a straightforward discrete grid enumeration, using a step size of two units. Note that enumerating all possible allocation combinations is time consuming because our problem is NP hard,

Table 2
Overview of the *DE* schemes and parameters

Scheme	Setting 1 DE/current-to-rand/1	Setting 2 DE/current-to-rand/1	Setting 3 DE/rand/1/bin
F	0.8	0.9	0.9
K	0.5	0.2	—
Cr	1	1	0.8
	Case 1	Case 2	
D	15	33	
NP	750	1650	

which means that the computational time increases in a polynomial way when adding a node. Therefore, only Case 1 can be enumerated within an appropriate time limit and the enumeration is restricted to feasible combinations while returning and delivering as much products as possible. Note that the *DE* algorithm does not receive this additional information. As a result, a lower bound (LB) for the optimal solution is found.

A first important observation for Case 1 from Table 3 is that the product flows according to the three *DE* settings converge to almost the same value, which in addition to this approximate to the flows found by enumeration. Furthermore, the profit according to the three *DE* solutions exceeds the LB. Since all the flows and the profit in Case 2 also converge to the same value according to the three *DE* settings (see Table 4), we are confident that the solution found by *DE* is close to the global optimum, although a LB could not have been determined. Next, whereas Tables 3 and 4 always show an alternative flow distribution among the customer locations for the *MINLP*-model, exactly the same demand is satisfied regardless of the *DE* settings. Another interesting result is that solving the same problem without accounting for inventory costs and uncertainty (i.e. the *MILP*-model) always leads to a network layout different from the *MINLP* solution.

With respect to the computation time, setting 1 is always much faster, which confirms the expectation of a slower convergence for settings 2 and 3 with $NP = 50D$. As an alternative, setting 3 with $NP = 5D$ found the same continuous flow variables for Cases 1 and 2 in considerably less time, respectively in 34 200 and 56 400 s. This reveals that the computation time for *DE* is strongly dependent on the parameter settings and that Case 1 can be solved faster by *DE* than by enumeration, even though a very limited number of points have been enumerated.

Evidently, computation time and memory requirements increase with the complexity of the network. Solving larger instances on a stand-alone pc offers the opportunity to do some preprocessing to find the best candidate network configuration. The main idea is to initially run the algorithm several times with a small population size to find the best layout fast, followed by an in depth investigation using the algorithm again but with the binary variables fixed to that best candidate. This reduces the search space, which enables the use of larger populations again that will guide us towards the solution. This procedure is applied to a case with three return markets, three reuse markets and initially nine potential facility locations that can be installed at three capacity levels. The size of this problem has practical relevance [6]. Similar results and conclusions as in Cases 1 and 2 were found.

Table 3
Computational results ‘Case 1’

Variables			*MILP*	*MINLP*			
			LINDO	*LB*	*DE*		
					Setting 1	Setting 2	Setting 3
X_{n_i}	Supply (n) $\rightarrow$ Facility (i)						
	1	1	50	14	14.91	14.91	14.05
	1	2	0	34	34.11	34.01	34
	1	3	10	12	10.98	11.09	11.94
	2	1	0	17	17.02	17.00	17.92
	2	2	0	0	0.001	0.03	0.003
	2	3	35	18	17.98	17.97	17.07
X_{i_n}	Facility (i) $\rightarrow$Demand (n)						
	1	1	30	0	15.68	19.73	8.36
	1	2	15	27.9	13.06	8.98	20.42
	2	1	0	15	12.61	7.76	1.18
	2	2	0	15.6	18.09	22.87	29.43
	3	1	0	15	1.72	2.5	20.46
	3	2	40.5	12	24.35	23.65	5.66
$X_i(q)$	Facility (i)-Capacity (q)						
	1	2	50	31	31.93	31.97	31.97
	2	2	0	34	34.11	33.27	34.003
	3	2	45	30	28.96	29.75	29.01
$\sum_i X_{i_n}$	Satisfied demand (n)						
		1	30	30	30	29.99	30
		2	55.5	55.5	55.5	55.5	55.51
Profit in euro			6096.4	5081.66	5083.53	5083.38	5083.35
Number of generations					1800	6000	24400
Computation time (s)				196990	18600	55320	248880

5. Conclusion

In this study, we have extended a facility-location *MILP*-model in a reverse logistics context with queueing relationships in order to incorporate a product’s cycle time and inventory holding costs, as well as to deal with the higher degree of uncertainty and congestion, typical characteristics of these networks. This extension is particularly relevant for time sensitive products with a high value. In order to search for the optimal solution of this *MINLP*-model, we proposed an algorithm based on the technique of differential evolution. Two main conclusions can be drawn from the investigation of several cases with increasing complexity. In the first place, the differential evolution finds robust solutions within an acceptable time limit that are very close to the true global optimum since their profit exceeds the lower

Table 4
Computational results 'Case 2'

Variables			MILP	MINLP			
			LINDO	LB	DE		
					Setting 1	Setting 2	Setting 3
	Supply (n) → Facility (i)						
X_{n_i}	1	1	45	—	15.96	15.93	15.95
	1	2	15	—	44.04	44.07	44.04
	1	3	0	—	0	0	0
	2	1	5	—	30.23	30.26	30.24
	2	2	0	—	4.77	4.74	4.76
	2	3	30	—	0	0	0
X_{i_n}	Facility (i) → Demand (n)						
	1	3	3	—	24.14	10.96	11.58
	1	4	42	—	17.44	30.61	29.99
	2	3	0	—	5.86	19.04	18.42
	2	4	13.5	—	38.06	24.89	25.51
	3	3	27	—	0	0	0
	3	4	0	—	0	0	0
$X_i(q)$	Facility (i)-Capacity (q)						
	1	1	0	—	0	0	0
	1	2	50	—	0	0	0
	1	3	0	—	46.19	46.19	46.19
	2	1	15	—	0	0	0
	2	2	0	—	0	0	0
	2	3	0	—	48.81	48.81	48.8
	3	1	30	—	0	0	0
	3	2	0	—	0	0	0
	3	3	0	—	0	0	0
$\sum_i X_{i_n}$	Satisfied demand (n)						
		3	30	—	30	30	30
		4	55.5	—	55.5	55.5	55.5
Profit in euro			6111.6	—	5053.75	5053.74	5053.75
Number of generations					1300	2900	8000
Computation time (s)					51 480	104 400	336 000

bound found by discrete grid enumeration and the population always converges to a similar solution. Secondly, compared to the traditional *MILP* solution, another network layout becomes more profitable when the *MINLP*-model is applied because various sources of uncertainty influence the location decision. Future research will be concerned with extending this new and improved model towards multiple products and multiple levels. Incorporating the queueing effects caused by transportation is also considered as a viable option.

Appendix A

Input data for the examples in Section 4.3 are given in Tables 5–12

Table 5
Yearly supply quantity and variability

	Returns	*SCV*
$n1$	60	1.5
$n2$	35	1.5

Table 6
Yearly demand quantity and market price in euro

	Demand	Price
$n3$	30	100
$n4$	60	80

Table 7
Potential facility locations and their yearly capacity levels

	$i1$	$i2$	$i3$
$q1$	30	30	30
$q2$	50	50	50
$q3$	60	60	60

Table 8
Yearly fixed costs for various capacity levels in euro

	$i1$	$i2$	$i3$
$q1$	200	150	225
$q2$	400	300	450
$q3$	500	385	575

Table 9
Variable costs/unit for various capacity levels in euro

	$i1$	$i2$	$i3$
$q1$	0.55	0.40	0.30
$q2$	0.45	0.35	0.25
$q3$	0.35	0.30	0.20

Table 10
Transportation cost/unit in euro

	$i1$	$i2$	$i3$
$n1$	6	10	6.25
$n2$	2.4	9	2.55

Table 11
Penalty costs/unit in euro

	Unsatisfied returns	Unsatisfied demand
$n1, n2$	27.50	—
$n3, n4$	—	0.50

Table 12
Disposal cost/unit in euro, disposal fraction, inventory cost/unit/year in euro and process time variability

	DC	γ	*IC*	*SCV*
$i1$	0.50	0.9	73	3
$i2$	0.25	0.9	54.75	2
$i3$	0.75	0.9	91.25	3.5

Appendix B

Finding an expression for the *SCV* of the arrivals at facility i $\left(c_{\mathrm{a}}^2(i)\right)$ based on the information about the *SCV* of the arrivals at the collection location n $\left(c_{\mathrm{a}}^2(n)\right)$ is not trivial because the stream at location n departs to various facilities i. Therefore define [27]

$f(n,i)$	proportion of products leaving disposer market n and going to facility i,
$c_{\mathrm{a}}^2(n,i)$	*SCV* of the interarrival time at facility i coming from disposer market n,
$c_{\mathrm{d}}^2(n)$	*SCV* of the departure time at disposer market n.

We have

$$f(n,i)=\frac{\lambda_{n_i}}{\lambda_n}.$$

The relation between $c_a^2(n,i)$ and $c_d^2(n)$ has been proven in the literature [10] and is given by

$$c_a^2(n,i)=f(n,i)c_d^2(n)+(1-f(n,i)).$$

In this case, the $c_d^2(n)$ and $c_a^2(n)$ are the same because firstly, $c_a^2(n)$ represents the variability of the collected return fraction and all collected goods have to be shipped to a reprocessing facility so there is no yield loss and secondly, no processing is required at collection location n so the variability is not changed.

Since a facility i can be supplied by different sources, its aggregate *SCV* of the arrivals $c_a^2(i)$ is the sum of a weighted average of the *SCV* of the arrivals at this facility i coming from the disposer markets n $c_a^2(n,i)$:

$$c_a^2(i)=\sum_n \frac{\lambda_{n_i}}{\lambda_i}\,c_a^2(n,i). \tag{20}$$

A similar approach can be found in [28]. All this information leads to

$$c_a^2(i)=\sum_n \frac{\lambda_{n_i}}{\lambda_i}\left(\frac{\lambda_{n_i}}{\lambda_n}\,c_a^2(n)+\left(1-\frac{\lambda_{n_i}}{\lambda_n}\right)\right).$$

Using the notation of the *MILP* network design model, this is equivalent to

$$c_a^2(i)=\sum_n \frac{X_{n_i}}{\sum_q X_i(q)}\left(\frac{X_{n_i}}{\sum_i X_{n_i}}\,c_a^2(n)+\left(1-\frac{X_{n_i}}{\sum_i X_{n_i}}\right)\right). \tag{21}$$

Appendix C

$$\phi(\rho,c_a^2,c_e^2,m)=\begin{cases}\left(\frac{4(c_a^2-c_e^2)}{4c_a^2-3c_e^2}\right)\phi_1(m,\rho)+\left(\frac{c_e^2}{4c_a^2-3c_e^2}\right)\psi(c^2,m,\rho), & c_a^2\geqslant c_e^2,\\ \left(\frac{c_e^2-c_a^2}{2(c_a^2+c_e^2)}\right)\phi_3(m,\rho)+\left(\frac{c_e^2+3c_a^2}{2(c_a^2+c_e^2)}\right)\psi(c^2,m,\rho), & c_a^2\leqslant c_e^2,\end{cases}$$

where

$$\phi_1(m,\rho)=1+\gamma(m,\rho),$$

$$\phi_3(m,\rho)=(1-4\gamma(m,\rho))\mathrm{e}^{-\frac{2(1-\rho)}{3\rho}},$$

$$\psi(c^2, m, \rho) = \begin{cases} 1, & c^2 \geqslant 1, \\ \phi_4(m, \rho)^{2(1-c^2)}, & 0 \leqslant c^2 \leqslant 1, \end{cases}$$

$$\text{with } \gamma(m, \rho) = \min\left\{0.24, \frac{(1-\rho)(m-1)\left[(4+5m)^{0.5}-2\right]}{16m\rho}\right\},$$

$$c^2 = \frac{c_{\mathrm{a}}^2 + c_{\mathrm{e}}^2}{2},$$

$$\phi_4(m, \rho) = \min\left\{1, \frac{\phi_1(m, \rho) + \phi_3(m, \rho)}{2}\right\}.$$

References

[1] Guide VDR, Jayaraman V, Srivastava R, Benton WC. Supply-chain management for recoverable manufacturing systems. Interfaces 2000;30(2):125–42.

[2] Babu BV, Angira R. A differential evolution approach for global optimization of MINLP problems. Proceedings of fourth Asia-Pacific conference on simulated evolution and learning, vol. 2, 2002. p. 880–4.

[3] Marianov V, ReVelle C. The queueing maximal availability location problem: a model for the siting of emergency vehicles. European Journal of Operational Research 1996;93:110–20.

[4] Marianov V, Serra D. Probabilistic, maximal covering location-allocation models for congested systems. Journal of Regional Science 1998;38:401–24.

[5] Amiri A. The design of service systems with queueing time cost, workload capacities and backup service. European Journal of Operational Research 1998;104:201–17.

[6] Krikke HR. Recovery strategies and reverse logistic network design. Disertation, University of Twente, The Netherlands; 1998.

[7] Fleischmann M. Quantitative models for reverse logistics. Disertation, Erasmus University Rotterdam, The Netherlands; 2000.

[8] Hopp WJ, Spearman ML. Factory physics. New York: McGraw-Hill; 2000.

[9] Whitt W. Approximations for the $GI/G/m$ queue. Production and Operations Management 1993;2(2):114–61.

[10] Shanthikumar JG, Buzacott JA. Open queueing network models of dynamic job shops. International Journal of Production Research 1981;19(2):255–66.

[11] Bitran GR, Tirupati D. Multiproduct queueing networks with deterministic routing: decomposition approach and the notion of interference. Management Science 1988;34(1):75–100.

[12] Ryoo HS, Sahinidis NV. Global optimization of nonconvex nlps and minlps with applications in process design. Computers & Chemical Engineering 1995;19(5):551–66.

[13] Ryoo HS, Sahinidis NV. A branch-and-reduce approach to global optimization. Journal of Global Optimization 1996;8(2):107–38.

[14] Adjiman CS, Androulakis IP, Floudas CA. Global optimization of minlp problems in process synthesis and design. Computers & Chemical Engineering 1997;21:S445–50.

[15] Androulakis IP, Maranas CD, Floudas CA. Alpha bb: a global optimization method for general constrained nonconvex problems. Journal of Global Optimization 1995;7(3):337–63.

[16] Kesavan P, Barton PI. Generalized branch-and-cut framework for mixed-integer nonlinear optimization problems. Computers & Chemical Engineering 2000;24:1361–6.

[17] Vaidyanathan R, El-Halwagi M. Global optimization of nonconvex minlps by interval analysis. In: Grossmann I, editor. Global optimization in engineering design. Dordrecht: Kluwer Academic Publishers; 1996. p. 175–93.

[18] Westerlund T, Skrifvars H, Harjunkoski I, Pörn R. An extended cutting plane method for a class of non-convex minlp problems. Computers & Chemical Engineering 1998;22(3):357–65.

[19] Storn R, Price K. Differential evolution—a simple end efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization 1997;11:341–59.

[20] Wah BW, Wang T. Efficient and adaptive lagrange-multiplier methods for nonlinear continuous global optimization. Journal of Global Optimization 1999;14:1–25.
[21] Rajasekaran S. On simulated annealing and nested annealing. Journal of Global Optimization 2000;16:43–56.
[22] Mathur M, Karale SB, Priye S, Jayaraman V, Kulkarni B. Ant colony approach to continuous function optimization. Industrial and Engineering Chemistry Research 2000;39:3814–22.
[23] Chelouah R, Siarry P. Tabu search applied to global optimization. European Journal of Operational Research 2000;123: 256–70.
[24] Salcedo RL. Solving nonconvex nonlinear programming problems with adaptive random search. Industrial & Engineering Chemistry Research 1992;31:262.
[25] Price KV. An introduction to differential evolution. In: Corne D, Dorigo M, Glover F, editors. New ideas in optimization. London: McGraw-Hill; 1999. p. 79–108.
[26] Lampinen J, Zelinka I. Mechanical engineering design optimization by differential evolution. In: Corne D, Dorigo M, Glover F, editors. New ideas in optimization. London: McGraw-Hill; 1999. p. 127–46.
[27] Lambrecht MR, Ivens PL, Vandaele NJ. ACLIPS: a capacity and lead time integrated procedure for scheduling. Management Science 1998;44(10):1548–61.
[28] Vandaele N, Vannieuwenhuyse I, Cupers S. Optimal grouping for a nuclear magnetic resonance scanner by means of an open queueing model. European Journal of Operational Research 2003;151(1):181–92.