

Optimal control of a distributed service system with moving resources: Application to the fleet sizing and allocation problem

Peter Köchel, Sophie Kunze, and Ulf Nieländer
Faculty of Computer Science
Chemnitz University of Technology
09107 Chemnitz, Germany
pko@informatik.tu-chemnitz.de

ABSTRACT

Usually two control problems are considered for a service system: Firstly, how to decide whether to accept or to reject an arriving client, and secondly, how many resource units to install. Today's renting firms offer the possibility to return a rented unit to an arbitrary location of the firm. Thus, a third control problem arises: How to organise transshipments of resource units between the locations of the firm. In our paper we pursue two aims. At first, we develop a queueing network model for the movement of units through the locations without any control. Such a model is helpful to find out the main streams of units as well as the locations with a significant imbalance between the inflow of units and the demand. At second, we give some answers to the second and third control problem under a given cost and gain structure. Our approach combines simulation with Genetic Algorithms (simulation optimisation). Some examples demonstrate the applicability of the proposed approach.

KEYWORDS

Fleet sizing and allocation problem, queueing network model, simulation, Genetic Algorithms

1. Introduction

In the past, usually two control problems with respect to a service system were considered: The first, how to decide whether to accept or to reject an arriving client (e.g. /11/), and the second, how many servers (resource units) to install (cp. /17/). Practical needs of modern transportation networks lead to a third control problem: How to organise transshipments of resource units between the nodes of the network. For instance, today's car rental firms offer the possibility to return a hired unit to an arbitrary location of the firm. As a consequence, a significant imbalance between the inflow of hired units and the demand arises for most of the locations. To overcome these imbalances the natural flow of units has to be corrected by reallocations. In scientific literature on design and control of transportation networks, the combination of the second and third problem is known as the *fleet sizing and allocation problem* (FSAP). Solving the FSAP requires answering two closely connected questions: How many units should a vehicle fleet contain (the *fleet sizing problem*), and how to redistribute empty vehicles that are not used in a given location among the locations of the network (the *empty vehicle reallocation problem*). A vehicle in our sense is a reusable unit for the realisation of a given kind of transportation service. Such units may be cars, trucks, railcars or airplanes as well as containers and material handling equipment.

For both the fleet sizing problem and the empty vehicle reallocation problem, voluminous literature exists (cp. /2/, /5/, /6/, and the references within). However, only a few papers consider the FSAP as the combination of the two problems /7/, /12/. That is the consequence of the complexity of the problem – we have to consider a set of interdependent locations – and of two inherent properties of transportation networks – the dynamic and uncertain character of the demand as well as of the performance of the network. The approaches to

solve the FSAP or at least one of its partial problems extend from linear and non-linear network programming models (e.g. /18/) via inventory and queueing theory models (cp. /7/, /10/) to dynamic programming models (see for instance /5/, /12/, /16/). However, no closed-form solutions are available. The majority of papers deals with algorithms that define an approximate solution for the discrete-time case with known demand and infinite transportation capacity.

In the present paper we pursue two aims. At first in Section 2, we develop a queueing network that models the movement of rented units through the locations without any control. Such a model is helpful to find out the main streams of rented units as well as the locations with a significant imbalance between the inflow of rented units and the demand. At second, we give some solutions for the FSAP. Because we cannot expect to solve the FSAP by any analytical approaches we use simulation optimisation. Both, our approach to combine simulation and optimisation and the simulator used and developed at Chemnitz University of Technology, are explained in Section 3. In Section 4 the results of some examples are discussed. The paper is finished by a short outlook.

2. Problem formulation and models

In this section we present both the mathematical formulation of the FSAP and some continuous time stochastic models. For the models considered here, we need at least the four element classes location, vehicles, clients, and transporters (to reallocate vehicles). These element classes are described next, prior to the mathematical formulation of the problem.

2.1. Elements of the model

A *location* is a defined place in a given region where vehicles can be stored, hired, and returned. Each location is characterised by a set of parameters and attributes. Parameters may be lower and upper levels of available vehicles, capacity of the queue of waiting clients, cost for holding a vehicle and rejecting a client as well as the gain for a hired vehicle. Additionally there may be parameters related to the applied reallocation policy. Attributes are for instance the numbers of available vehicles, of waiting clients, or the amount of vehicles, which are in transport to the given location.

The *vehicles* represent the moveable resources in the model. Each of a finite number of vehicles can be in one of the three states waiting (or ready for using), in use (hired), and in transport.

The *clients* are temporary elements of the model, i.e., they enter the system at a given location and leave it after using a vehicle or after being rejected. Clients will be characterised by their arrival and hiring behaviour. That behaviour describes the demand or the load of the system.

Transporters are transshipment units (e.g. railroad car-racks or special trucks) for the reallocation of vehicles. Each of them has a finite capacity and needs a random transshipment time between a given pair of locations. Transportation costs depend on quantity and duration of the transshipment.

2.2. The optimisation problem

The decision variables of the FSAP are the fleet size K , the number of vehicles available in the system, and the reallocation policy π . The fleet size is an integer variable, i.e., $K \in \mathbb{N} = \{1, 2, \dots\}$. Concerning the reallocation policy π we will restrict ourselves to a given set Π of admissible policies (cp. subsection 2.4). If $g(K, \pi)$ denotes the gain per time unit in the steady state for given K and π , solving the FSAP is equivalent to solving the maximisation problem

$$(P-1) \quad g(K, \pi) \quad \rightarrow \quad \underset{(K, \pi) \in N \times \Pi}{\text{MAX}} !$$

We will investigate (P-1) under the following assumptions:

- (1) Time is continuous.
- (2) Clients arrive at a given location from the exterior of the system following a Poisson process.
- (3) All vehicles are homogeneous without failures, damage, and they don't get old.
- (4) Hired vehicles will be used for a random time and returned at a randomly chosen location.
- (5) There exists an infinite number of transporters to tranship vehicles between locations.
- (6) The transporters are homogeneous; they have a finite capacity, and they need a random time for a given transport.
- (7) All random variables (r.v.) are independent.

Next, we define an analytical expression for the goal function g . Let M denote the number of locations in the network. For each location i , $i = 1, 2, \dots, M$, let denote:

- λ_i - the arrival intensity of clients in location i ;
- QC_i - the queueing capacity of waiting clients in location i ;
- g_i - the gain per time unit for a vehicle rented in location i ;
- wv_i - the waiting cost per time unit and vehicle available (not rented) in location i ;
- wc_i - the waiting cost per time unit and client waiting for a vehicle in location i ;
- r_i - the rejection cost per rejected (lost) client in location i .

Furthermore, let TC denote the capacity of the transporters. To describe the randomness we need the following definitions:

- UT_{ij} - r.v. "using time of a vehicle rented in i and willing to be returned in j ";
- TT_{ij} - r.v. "time for transporters to go from i to j ";
- p_{ij} - the probability that a vehicle rented in i will be returned in j .

Finally, we need some steady-state performance measures:

- H_i - the mean number of vehicles hired per time unit in location i ;
- W_i - the mean number of waiting vehicles in location i ;
- Q_i - the mean number of waiting clients in location i ;
- R_i - the mean number of per time unit rejected clients in location i ;
- T_{ij} - the mean number of per time unit realised transhipments from i to j .

With these notations the goal function g of (P-1) may be expressed as

$$g(K, \pi) = \sum_{i=1}^M [g_i \cdot H_i - wv_i \cdot W_i - wc_i \cdot Q_i - r_i \cdot R_i] - C(K, \pi), \quad (2.1)$$

where $C(K, \pi)$ denotes the expected cost per time unit in the steady state for a fleet size K in connection with the reallocation policy π . We remark that all other performance measures in (2.1) are as well complex functions of both the parameters of the system and the decision variables K and π . Since $\lambda_i = H_i + R_i$ or $H_i = \lambda_i - R_i$ holds for all locations $i = 1, 2, \dots, M$, we can write

$$g(K, \pi) = \sum_{i=1}^M (g_i \cdot \lambda_i) - f(K, \pi), \quad (2.2)$$

with

$$f(K, \pi) = \sum_{i=1}^M [wv_i \cdot W_i + wc_i \cdot Q_i + (r_i + g_i) \cdot R_i] + C(K, \pi). \quad (2.3)$$

From equations (2.2) and (2.3) one can see that the maximisation of goal function g is equivalent to the minimisation of function f . Equation (2.2) states the intuitive fact that the gain is equal to the reward from serving *all* the arriving clients, the load of the system, minus the cost for waiting vehicles, waiting clients, rejected clients, and reallocations of vehicles.

Here, the rejection cost includes as well the original rejection cost as the lost reward. Consequently, instead of solving problem (P-1) we can also solve problem

$$(P-2) \quad f(K, \pi) \quad \rightarrow \quad \text{MIN !} \\ (K, \pi) \in N \times \Pi$$

Most of the complications for solving the FSAP are caused by the term $C(K, \pi)$ for given K and policy $\pi \in \Pi$. Since we cannot expect to get an analytically tractable expression for $C(K, \pi)$ we can give only an approximate solution for the FSAP. For that reason we will tackle the problem in two steps. In the first step we restrict the set Π to one element, to policy π_0 that realises no reallocations at all. In this case the term $C(K, \pi)$ can be omitted in (2.1) or (2.3). The resulting optimisation problem is the *fleet sizing problem without vehicle reallocation*. It will be considered in the next subsection. In a second step we restrict ourselves to sets of policies with a simple but intuitive efficient structure, as for instance (s, Q) -policies in inventory theory. Such policies are characterised by a few parameters. By choosing optimal values for these parameters, we hope to get a nearly optimal solution for the FSAP. This will be done in Section 3.

2.3. Models for the fleet sizing problem without vehicle reallocation

In the present subsection we assume that no reallocations are possible, i.e., $\Pi = \{\pi_0\}$. We further assume $QC_i = 0$ for $i = 1, 2, \dots, M$, i.e., demand which cannot be satisfied by an available vehicle will be lost (the *no-backorder case*). Finally, the using time UT_{ij} of a vehicle rented in location i and willing to be returned in location j is assumed to be a Coxian distributed random variable with expected value μ_{ij} , $i, j = 1, 2, \dots, M$ (see e.g. /3/). With these additional assumptions the entire system can be modelled as a closed queueing network with K jobs (see Fig. 1). The network contains $M+M^2$ nodes – node 1 to M for the locations, and M^2 artificial nodes $a(i, j)$ representing that a vehicle is used by renting it at location i , and that it will be returned in location j ($i, j = 1, 2, \dots, M$). The vehicles are the jobs circulating through the network with transition probabilities $p_{i a(i, j)} = p_{ij}$, $p_{a(i, j) j} = 1$, and $p_{ij} = 0$ else, $i, j = 1, 2, \dots, M$. Furthermore, from the above assumptions it follows that the network is a BCMP queueing network (cp. /3/) with Type-1 nodes i , i.e. i is a $\cdot/M/1$ -node with service intensity λ_i , and Type-3 nodes $a(i, j)$, i.e. $a(i, j)$ is a $\cdot/G/\infty$ -node with service intensity μ_{ij} , $i, j = 1, 2, \dots, M$. The state of that network is defined as a $(M+M^2)$ -dimensional vector whose components represent the number of vehicles in the different nodes.

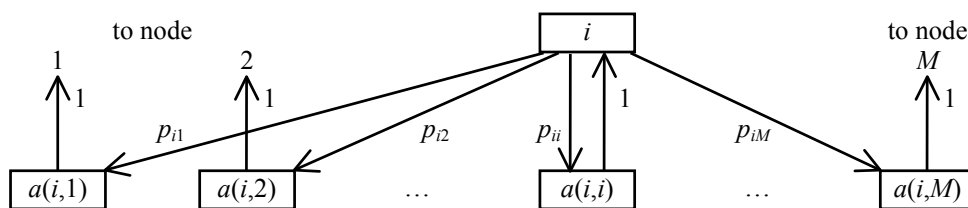


Figure 1. The queueing network for the fleet sizing problem

We remark that for closed queueing networks the steady state probabilities always exist. Let $p_i(k)$ and $p_{a(i, j)}(k)$, $k = 0, 1, \dots, K$, $i, j = 1, 2, \dots, M$, denote the corresponding steady state probabilities for the considered network. Then for each node i of the network, the needed steady state performance measures can be expressed by these probabilities in the following way:

- (1) Mean number H_i of vehicles hired per time unit

$$H_i = \sum_{j=1}^M H_{ij} \quad \text{with} \quad H_{ij} = \sum_{k=1}^K k \cdot p_{a(i,j)}(k), \quad i, j = 1, 2, \dots, M;$$

(2) Mean number W_i of waiting vehicles

$$W_i = \sum_{k=1}^K k \cdot p_i(k), \quad i = 1, 2, \dots, M;$$

(3) Mean number R_i of per time unit rejected clients

$$R_i = \lambda_i \cdot p_i(0), \quad i = 1, 2, \dots, M.$$

For the corresponding global performance measures, i.e. the expected number H of hired vehicles per time unit, the expected number W of waiting vehicles, and the expected number R of rejected clients in the system, it simply holds

$$H = \sum_{i=1}^N H_i, \quad W = K - H = \sum_{i=1}^N W_i, \quad \text{and} \quad R = \sum_{i=1}^N R_i.$$

It is important to remark that the steady-state probabilities as well as the performance measures depend on the number K of vehicles. For simplicity we do not express it here. Since we want to maximise the return earned from the vehicle fleet, the criterion function of (2.1) should be written as

$$g(K) := g(K, \pi_0) = \sum_{i=1}^M [g_i \cdot H_i - wv_i \cdot W_i - r_i \cdot R_i]. \quad (2.4)$$

For a given fleet size K , function $g(\cdot)$ represents the average reward per time unit in the steady state. The optimisation problem (P-1) is reduced to problem

$$(P-3) \quad \{\text{Define } K^* \in N: g(K^*) \geq g(K) \text{ for all } K \in N.\}$$

To solve (P-3) we can use the following greedy algorithm:

Initialisation: Set $K = 0$; compute $g(0) = -\sum_{i=1}^M r_i \cdot \lambda_i$.

Iteration: REPEAT
 $K := K+1$;
 Compute $g(K)$
 UNTIL $g(K) < g(K-1)$.

Output: $K^* := K-1$; $g(K^*) = g(K-1)$.

Applying that greedy algorithm, two questions arise: First, how to compute $g(K)$, and second, why should the output K^* be the optimal fleet size?

The answer to the first question is given by the following adaptation of the well-known *Mean Value Algorithm* (MVA) to the present model. For BCMP-networks the MVA computes the three performance measures mean number of jobs in each node, mean response time V of a job in each node, and the throughput Th of the network (see e.g. /3/). A necessary input for the MVA are the so-called *visit frequencies* to all nodes. These frequencies describe the mean number of visits to a node between two consecutive visits in a given reference node, e.g. node 1. For the investigated network these frequencies are the solution of the following system of linear equations:

$$\begin{cases} e_1 = 1; \\ e_i = \sum_{j=1}^M e_{a(j,i)}, \quad i = 2, 3, \dots, M; \\ e_{a(i,j)} = e_i \cdot p_{ij}, \quad i, j = 1, 2, \dots, M. \end{cases} \quad (2.5)$$

To describe the MVA let $X(n)$ denote the value of performance measure X for a closed BCMP-network with n clients.

MVA for the Fleet Sizing Problem:

Input: $K; \{e_i, e_{a(i,j)}, i, j = 1, 2, \dots, M\}.$

Initialisation: $H_{ij}(0) = 0, W_i(0) = 0, i, j = 1, 2, \dots, M.$

Iteration: For $n = 1, 2, \dots, K$ do

a)
$$\begin{cases} V_i(n) = 1/\lambda_i \cdot [1 + W_i(n-1)], & i = 1, 2, \dots, M; \\ V_{a(i,j)} = 1/\mu_{a(i,j)}, & i, j = 1, 2, \dots, M; \end{cases}$$

b)
$$Th(n) = n / \sum_{i=1}^M \left[e_i \cdot V_i(n) + \sum_{j=1}^M e_{a(i,j)} \cdot V_{a(i,j)}(n) \right];$$

c)
$$\begin{cases} W_i(n) = Th(n) \cdot e_i \cdot V_i(n), & i = 1, 2, \dots, M; \\ H_{ij}(n) = Th(n) \cdot e_{a(i,j)} / \mu_{a(i,j)}, & i, j = 1, 2, \dots, M. \end{cases}$$

For the general model of Fig.1 there exist at least three interesting special cases:

- a) Case “ $UT_{ij} = UT_i$ ”, i.e., the using time of a vehicle depends only on the location the vehicle was rented in. In this case the M^2 artificial nodes in Fig.1 can be reduced to M artificial nodes $a(i), i = 1, 2, \dots, M$. For the transition probabilities $p_{i a(i)} = 1, p_{a(i)j} = p_{ij}, i, j = 1, 2, \dots, M$, holds.
- b) Case “ $UT_{ij} = UT_j$ ”, i.e., the using time of a vehicle depends only on the location the vehicle will be returned to. Again, instead of M^2 artificial nodes we now need M nodes $a(i)$ only with $p_{j a(i)} = p_{ji}, p_{a(i)i} = 1, i, j = 1, 2, \dots, M$.
- c) Case “ $p_{ij} = p_j$ and $UT_{ij} = UT$ ”, i.e., the returning probabilities of a vehicle depend only on the location the vehicle will be returned to, and the using time is independent of the locations. The resulting network can be modelled as a Central-server model (see /3/), in which one artificial node “0” with service intensity $\mu_0 := 1 / E(UT)$ represents the M^2 nodes $a(i,j)$ (cp. Fig. 2). In that simple special case of the general problem, the solution of (2.5) is equal to

$$e_0 = 1, e_i = p_i, i = 1, 2, \dots, M. \tag{2.6}$$

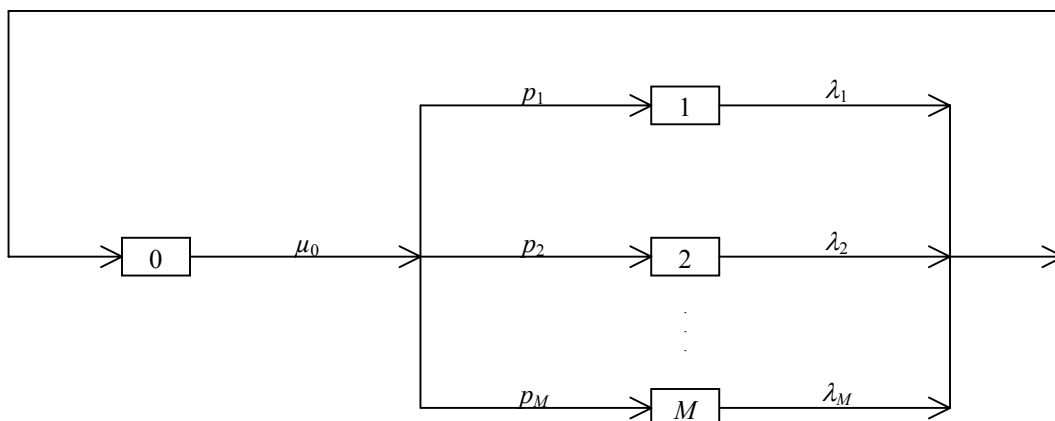


Figure 2. The Central-server model

In case c) the MVA for the fleet sizing problem becomes a very simple formulation.

MVA-II for the Fleet Sizing problem:

Input: K .

Initialisation: $W_i(0) = 0, i = 0, 1, \dots, M$.

Iteration: For $n = 1, 2, \dots, K$ do

a) $V_i(n) = 1/\lambda_i [1 + W_i(n-1)], i = 1, 2, \dots, M$.

b) $Th(n) = n / \left[1/\mu_0 + \sum_{i=1}^M p_i \cdot V_i(n) \right]$.

c) $\begin{cases} W_i(n) = Th(n) \cdot p_i \cdot V_i(n), & i = 1, 2, \dots, M; \\ W_0(n) = Th(n) / \mu_0. \end{cases}$

In MVA-II the steady-state performance measure $W_0(n)$ represents the mean number of hired vehicles for a fleet of size n . If g_i is constant for all locations i , then the goal function from (2.4) can be written as

$$g(K) = g \cdot W_0(K) - \sum_{i=1}^M [wv_i \cdot W_i(K) + r_i \cdot R_i(K)]. \quad (2.7)$$

If the gains for rented vehicles are different for different locations we cannot define the mean numbers H_i from the Central-server model. However, we can still use the MVA-II because of

$$H_i(K) = Th_i(K) / \mu_0 = Th(K) \cdot p_i / \mu_0 = W_0(K) \cdot p_i, \quad i = 1, 2, \dots, M. \quad (2.8)$$

Let us consider a numerical example for case c).

Example 2.1

Let the time be measured in days and let $E(UT) = 2$ days, i.e., $\mu_0 = 0.5/\text{day}$. For $M = 5$ locations we assume:

i	1	2	3	4	5
wv_i	4	4	4	4	1
r_i	4	4	4	4	15
λ_i	3	3	3	3	3
g_i	15	15	15	15	15
p_i	0.1	0.1	0.1	0.1	0.6

With respect to these data one can see that the locations 1 to 4 have identical characteristics. Furthermore, these locations will be shortage locations whereas location 5 will be a surplus location. The application of the greedy algorithm formulated above, with MVA-II for computing the performance measures needed, now returns the following data. We remark that $W_i(n)$ is identical for $i = 1, 2, 3, 4$ and all n .

n	0	1	2	3	4	5	6	7	8	9	10	11
$Th(n)$	0	0.4286	0.8502	1.2636	1.6674	2.0595	2.4379	2.7999	3.1425	3.4623	3.7558	4.0193
$W_0(n)$	0	0.8372	1.7004	2.5272	3.3348	4.1190	4.8758	5.5998	6.2850	6.9246	7.5116	8.0386
$W_1(n)$	0	0.0143	0.0287	0.0433	0.0580	0.0726	0.0872	0.1015	0.1154	0.1287	0.1413	0.1529
$W_5(n)$	0	0.0857	0.1846	0.2994	0.4333	0.5904	0.7755	0.9942	1.2534	1.5604	1.9233	2.3499
$g(n)$	-93.00	-75.91	-59.13	-42.69	-26.67	-11.14	3.81	18.06	31.49	43.95	55.30	65.39

n	12	13	14	15	16	17	18	19	20	21	22	23
$Th(n)$	4.2498	4.4451	4.6047	4.7296	4.8230	4.8892	4.9338	4.9623	4.9795	4.9893	4.9947	4.9975
$W_0(n)$	8.4996	8.8902	9.2094	9.4592	9.6460	9.7784	9.8676	9.9246	9.9580	9.9786	9.9894	9.9950
$W_1(n)$	0.1633	0.1724	0.1799	0.1860	0.1907	0.1940	0.1964	0.1979	0.1988	0.1994	0.1997	0.1998
$W_5(n)$	2.8472	3.4203	4.0708	4.9766	5.5914	6.4454	7.3468	8.2839	9.2754	10.2239	11.2119	12.2957
$g(n)$	74.08	81.29	87.00	91.25	94.17	95.95	96.82	97.02	96.74	96.15	95.38	94.49

n	24	25	26	27	28	29	30	31	32	33	34	35
$Th(n)$	4.9988	4.9995	4.9998	4.9999	5.000	5.000	5.000	5.000	5.000	5.000	5.000	5.000
$W_0(n)$	9.9976	9.9990	9.9996	9.9998	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0
$W_1(n)$	0.1999	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
$W_5(n)$	13.2026	14.2012	15.2005	16.2002	17.2001	18.2	19.2	20.2	21.2	22.2	23.2	24.2
$g(n)$	93.55	92.58	91.59	90.60	89.60	88.60	87.60	86.60	85.60	84.60	83.60	82.60

From these results for Example 2.1 one can draw some interesting conclusions. Obviously, the optimal fleet size is equal to 19 vehicles with a mean gain per day of 97.02 monetary units. One can see that the goal function $g(\cdot)$ is sufficiently flat nearby the optimal value for the fleet size. Thus fleet sizes between 17 and 22 are acceptable. But a fleet size of 13 yields only about 84 % of the optimum. Furthermore, for fleet sizes greater than 25 an increase of the fleet size results in a linear decrease of the goal function. This is caused by the fact that for the given example with a sufficiently large fleet size an additional vehicle on average will wait at location 5 (see the values for $W_5(n)$ for $n > 25$), where it will cause one unit of waiting cost. Thus, an overestimation of the optimal fleet size is less costly than an underestimation. The next observation is that the throughput $Th(n)$ and the expected number of used vehicles $W_0(n)$ converge to constant values. This is typical for service systems – above a given amount of service resources an additional service resource has no effect on the overall service level. A more careful analysis of the results shows that the throughput $Th(\cdot)$ as well as the gain function $g(\cdot)$ and the expected number of used vehicles $W_0(\cdot)$ are concave functions of the fleet size K . If we could prove the concavity property in general we would have the answer to the second question related to the greedy algorithm: The computed fleet size K^* is the optimal one. Unfortunately, up to now only the following results are available:

1. For closed networks with exponentially distributed service times (Jackson networks) both the throughput $Th(\cdot)$ of the network and the throughputs $Th_i(\cdot)$ of the locations are increasing and concave functions of the number K of jobs in the system (cp. Theorem 1.5.3 in [4]).
2. Since it holds for Type-3 nodes that the mean number of jobs in the node is equal to the node's throughput divided by the service rate, the increasing and concavity property of the throughput will be transferred to the means $H_{ij}(\cdot)$ and $W_0(\cdot)$.
3. The mean numbers $R_i(\cdot)$ of per time unit rejected clients are decreasing and convex functions of K due to the equality $R_i = \lambda_i - Th_i$ for $i = 1, 2, \dots, M$.

To prove the concavity of the goal function g from (2.4) or (2.7) it now remains to prove that the mean numbers $W_i(\cdot)$ of cars waiting in the locations $i = 1, 2, \dots, M$ are convex functions of K . However, the results for Example 2.1 show that convexity for $W_i(\cdot)$ does not hold in general. Thus function $W_1(\cdot)$ has an S-shape behaviour – convex for small and concave for

larger values of K . Nevertheless, we hope to prove the concavity for the goal function in the near future or, at least, to find sufficient conditions for concavity.

To finish the present subsection we remark, that the results on problem (P-3), the fleet sizing problem without reallocations, may be helpful for the FSAP. First, the value of K^* gives a certain indication for the search of the optimal fleet size for the FSAP. Second, the value $g(K^*)$ may serve as a lower bound for the optimal gain for the FSAP. And third, as additional information from the MVA it is possible to see which locations are surplus and which are shortage locations. That knowledge can be used to define good heuristic reallocation policies. In the next subsection we will define some meaningful reallocation policies for the FSAP.

2.4. Reallocation policies for the fleet sizing and allocation problem

If we would allow proper reallocation policies, i.e., if we would assume that $\Pi \neq \{\pi_0\}$, we could not expect to get an analytical solution for the FSAP. Therefore, we are looking for approximate solutions. We will make the approximation in the following sense. First, we restrict ourselves to classes of simple structured reallocation policies, which may not contain the true optimal reallocation policy. Second, computing corresponding values of the goal function is not possible. Thus we will estimate these values by a simulation experiment. Before we shortly describe the simulator for a transportation network in Section 3, we will discuss some classes of reallocation policies, which are simple-structured and which seem to be meaningful for the considered problem.

Because we assumed a continuous-time model, in a first step we restrict the set Π of all admissible reallocation policies to the basic class of *situation-dependent* policies Π_S . Situation-dependent means that a reallocation order will be triggered off by the occurrence of a defined situation in the system, as for instance the (s, Q) -policy in an inventory system. We consider two kinds of such situations – the *shortage*-situation when a location has a shortage of vehicles, and the *surplus*-situation when a location has too many vehicles. Thus we can define three subclasses of Π_S :

- (1) The set Π_1 of shortage-based reallocation policies,
- (2) set Π_2 of surplus-based reallocation policies, and
- (3) set Π_3 of shortage/surplus-based reallocation policies as a combination of (1) and (2).

A second classification aspect may be the number of locations, that are taking part in a given reallocation. If we assume continuous random variables then the following cases are relevant:

- (A) The 1-supplier-1-receiver case,
- (B) the 1-supplier-multi-receiver case, and
- (C) the multi-supplier-1-receiver case.

We remark that we have one receiver in a shortage-situation whereas we have one supplier in a surplus-situation. To find out the real situation in a given location i we introduce the notion of the *location balance* B_i as the sum of vehicles waiting in location i or being in transport to location i minus the number of clients waiting in location i . Now we can describe the implemented policies as follows.

- (1) Location i applies a (s_i, Q_i) -policy if it observes the following rule:
IF $B_i < s_i$ THEN place a demand order for $Q_i \geq 1$ vehicles.
- (2) Location i applies a (S_i, q_i) -policy if it observes the following rule:
IF $B_i > S_i$ THEN place a delivery order for $q_i \geq B_i - S_i$ vehicles.
- (3) Location i applies a (s_i, Q_i, S_i, q_i) -policy if it observes the following rule:
IF $B_i < s_i$ THEN place a demand order for $Q_i \geq 1$ vehicles ELSE
IF $B_i > S_i$ THEN place a delivery order for $Q_i \geq B_i - S_i$ vehicles.

In fact, in the currently available simulator (see /13/) all locations should follow the same rule, but the relevant parameter values may be different. For abbreviation we denote the implemented reallocation policies by $sQ_1, sQ_n, Sq_1, Sq_n, sS_1$, and sS_n , and in addition

to the parameters for the location policy the third parameter, 1 or n respectively, indicates how many suppliers or receivers can be involved in a reallocation.

3. The evolutionary optimisation approach to the FSAP

Let us give a short introduction to the approach employed at the Chemnitz University of Technology to solve extremely complex optimisation problems (cp. /1/, /9/, /14/). The approach combines simulation with evolutionary optimisation, i.e. Genetic Algorithms and Evolutionary Strategies. For an overview on simulation optimisation see e.g. /8/.

The general idea of our optimisation approach is outlined in Fig. 3. For a given decision problem an optimiser proposes a solution. To estimate the performance of that solution, results of a simulation experiment or/and results of analytical computations can be used. On the basis of the estimated performance the optimiser decides to accept the current solution or not. Acceptance stops whereas rejection continues the search process.

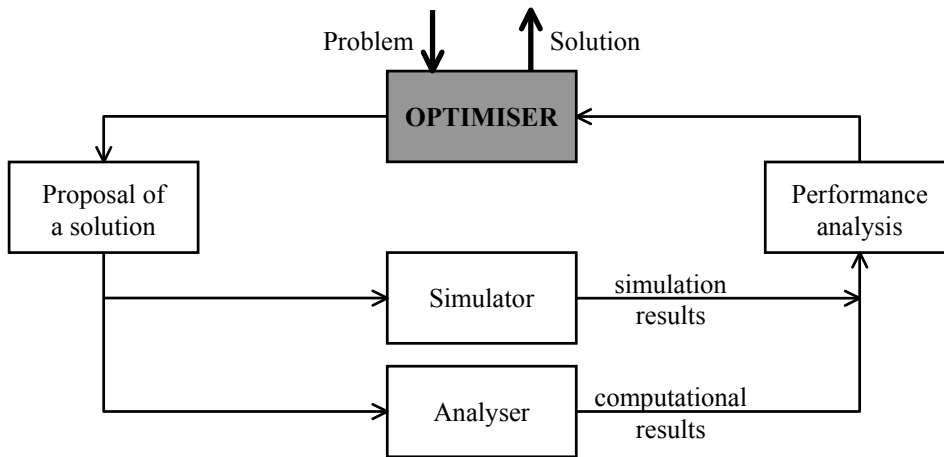


Figure 3. The principle of the used optimisation approach

In order to get a better insight we refine the general principle of Fig. 3 to the more detailed representation in Fig. 4. In Fig. 4 we denote an optimisation problem by the pair (f, D) , where f denotes the goal function, and D the set of admissible decisions. For an efficient realisation of the proposed optimisation approach the optimiser should comprise the four depicted components. The *starter* chooses an initial decision $x \in D$, which is presented to the simulator. The simulator realises a simulation experiment. With the simulation results the *estimator* generates an estimate for the value $f(x)$ of the criterion function. On the basis of that estimate (and may be further available information) the *decider* either accepts the currently investigated decision as approximately optimal or a *searcher* gets the order to find another decision. Clearly for a concrete application the components of the optimiser have to be implemented. We remark that by definition of appropriate interfaces the combination of in principle arbitrary simulators and optimisers is possible.

To apply the scheme of Fig. 4 to problem (P-1) we had to implement a simulator for transportation systems and to realise the building components of the optimiser. In view of the complexity of the considered problem we preferred Genetic Algorithms as optimisation approach since they can be designed to be very robust with respect to the random output of the simulation experiments for different decisions.

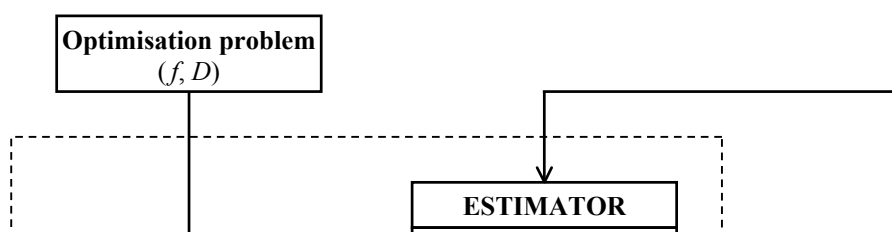


Figure 4. The scheme of the simulation optimisation

The applied Genetic Algorithm was taken from the evolutionary optimisation tool LEO (Laboratory for Evolutionary Optimisation), developed as well at our university (see /14/). More information about LEO can be found in /15/. To represent a decision within the optimisation process, LEO uses a number of chromosomes. Those chromosomes in general are strings (vectors) of bits (classic GA) or reals (real-coded GA) or integers (suitable for the FSAP) or permutations (for combinatorial optimisation problems). A chromosome for the FSAP comprises both the number K of vehicles and the parameters of the relevant reallocation policy. For instance, for $M = 5$ locations and policy sQ_1 the vector (66, 24, 20, 20, 18, 4, 6, 6, 6, 6, 21) represents such a chromosome with components $K = 66$, $s_1 = 24$, ..., $s_5 = 4$, $Q_1 = 6$, and so on. The optimisation process in LEO contains five steps. In step 0, the initialisation of the chromosomes is realised, i.e. the initial population of a given number of decisions' representations (chromosomes) is generated. This can be done by assigning random values to the components of a chromosome. In step 1, for each of the chromosome types of the representation one corresponding genetic operator is chosen. For instance, the one-point crossover takes two parent-chromosomes $x' = (K', s_1', \dots, s_M', Q_1', \dots, Q_M')$ and $x'' = (K'', s_1'', \dots, s_M'', Q_1'', \dots, Q_M'')$, randomly chooses an integer t , $1 \leq t \leq 2 \cdot M$, and, if e.g. t is between 2 and M , produces the two child-chromosomes $y' = (K', s_1', \dots, s_{t-1}', s_t'', \dots, s_M', Q_1'', \dots, Q_M'')$ and $y'' = (K'', s_1'', \dots, s_{t-1}'', s_t', \dots, s_M'', Q_1', \dots, Q_M')$. Apart from the classical crossover and mutation there exist special operators that can be applied to number chromosomes (as it is the case for the FSAP). Altogether nine operators for number chromosomes and eight for bit and permutation chromosomes are implemented in LEO. The application of such an operator takes some parent-chromosomes and produces some child-chromosomes. In step 2, the required number of parent-decisions are selected using the selection methods. They are all implemented both as better-selection to prefer decisions with better objective value and as worse-selection to prefer decisions with worse objective value. We remark that for the FSAP the objective function can be given by formula (2.1) or (2.3). In step 3, the child-decisions are created using the operators chosen in step 1. In step 4, the child-decisions are taken into the population by worse-selecting and deleting other decisions

from the population. To avoid premature convergence, no duplicate decisions are inserted into the population. The concept additionally includes elitism, i.e. the best decision found so far cannot be lost again. In LEO, also multiple isolated populations are possible with temporary migration of decisions from one population to the next. LEO also builds a family tree up to a certain depth of all generated decisions to record, which of the genetic operators were used to create that decision and which of selection methods were used to choose its parents before. This family tree can be evaluated to find out, which of the genetic operators and selection methods performed good so far on the actual optimisation problem by producing better decisions out of worse ones. The aim is to apply them more often to increase the efficiency of the optimisation process.

After we sketched out the optimisation process let us say some words about the simulator FSAS. The simulator FSAS (*F*leet-*S*ize-*A*llocation-*S*imulator), developed and available in a laboratory version at Chemnitz University of Technology, is written in C++ (cp. /13/). FSAS may be used for the simulation of transportation systems as described in Section 2. Using the event-oriented approach the simulator models the interactions of the four model elements locations, vehicles, clients, and transporters. To model random influences the simulator supports the use of exponential, uniform, and lognormal distributions at present.

The number of *locations* is an integer greater one. For each location the distribution of the inter-arrival times of clients, the queueing capacity for waiting clients (infinite, finite or zero), several gain and cost factors, and above all the applied reallocation policy should be defined.

The *vehicles*, whose number is constant during a given simulation experiment, are homogeneous. They have an infinite lifetime without accidents and breakdowns. No buying or selling actions are allowed. Capital costs of holding a vehicle in the system are included in the gain from vehicle renting and in the waiting cost for a free vehicle.

Clients are generated by the locations according to their inter-arrival distributions. For each client arriving at a given location both the returning probabilities and the corresponding using times of hired cars are defined.

The simulator FSAS assumes an infinite number of homogeneous *transporters*, i.e. transportation orders will be executed immediately. However, depending on the starting point and the point of destination, transportation will take a random time. All transporters have the same finite capacity. As the vehicles, the transporters have an infinite lifetime without accidents and breakdowns, too. Transshipments produce costs. Two cases are implemented – the linear case, where transportation cost arise per time unit and per transported vehicle, and the case, where in addition to the linear costs we have some set-up costs. For transshipments with more than one supplier or receiver, intermediate stops and additional loadings and un-loadings are possible.

Let us briefly describe how the transshipment routes are chosen in FSAS. At first, we consider the policies with one supplier and one receiver, as for instance the policy sQ_1 . To apply such a policy, for a given receiver we have to find one supplier out of a non-empty set of potential suppliers. The following (heuristic) definition was implemented for that reason: Location j is a potential supplier for location i if $B_j \geq Q_i + s_j + 1$, i.e., after the transshipment of Q_i vehicles out of location j the next arriving client to location j must not generate a demand order. In case the set of potential suppliers is empty there exists a first-come-first-served queue of backordered demand orders. The other reallocation policies are implemented in a similar way. To choose the one supplier out of the set of potential suppliers two heuristics are meaningful: Choosing the fastest supplier which is at the same time the cheapest one in case of only time-dependent transportation cost or choosing the richest supplier with the most available vehicles. The advantage of the second heuristic is that we have a better control on accumulations of vehicles in some locations; the disadvantage is that we have no control on the costs. Thus at present the first heuristic is implemented in FSAS, i.e., from the set of potential suppliers the one with the shortest average transshipment time will be chosen. This

heuristic is also used for finding an optimal transshipment in the case of one receiver and several suppliers respectively one supplier and several receivers. To define an optimal transshipment plan in a given situation, a recursive search algorithm is implemented in FSAS which in fact checks all routes through the system of locations to find the cheapest connection between the starting and destination points of a transshipment. This simple method works well since we have either one receiver or one supplier in all situations. In the present version of FSAS, all locations have to follow the same *reallocation policy*, but the policy parameters may be different. Therefore for a concrete simulation experiment we can choose from the policies $sQ_1, sQ_n, Sq_1, Sq_n, sS_1$, and sS_n . Furthermore we can choose the policy π_0 that allows no reallocations at all.

As output the simulator returns both estimations for different steady-state performance measures and the estimation for the goal function g from (2.1). Additionally, the estimation for the service level of the system, i.e. the rate of served clients, is returned.

4. Examples

To show the applicability of the proposed approach and to get some feeling about the quality of the introduced reallocation policies we carried out some runs with the above described software tools. In addition to these aims it is interesting to compare the solution for the FSAP, the fleet sizing and allocation problem, with the solution for the FSP, the fleet sizing problem without reallocation of empty vehicles. Consequently, we chose the system with five locations from Example 2.1 (see Fig. 5 for the layout).

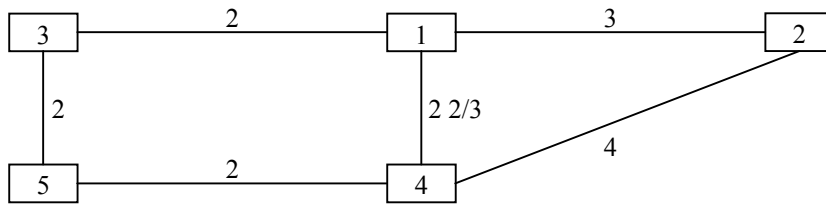


Figure 5. Layout of the exemplary system

In addition to the data of that example, which we repeat here for better understanding, we have to define the transportation times and the transportation costs for transporters. The transportation times are measured in hours [h]. We assume $TT_{ij} = TT_{ji}$ and the following uniform distributions $U_{ij}(a, b)$:

\times	$U_{12}(2.5, 3.5)$	$U_{13}(1.92, 2.08)$	$U_{14}(2.4, 2.933)$	$U_{15}(3.2, 4.8)$
$U_{21}(2.5, 3.5)$	\times	$U_{23}(4.8, 5.2)$	$U_{24}(3.2, 4.8)$	$U_{25}(5.96, 6.04)$
$U_{31}(1.92, 2.08)$	$U_{32}(4.8, 5.2)$	\times	$U_{34}(3.8, 4.2)$	$U_{35}(1.92, 2.08)$
$U_{41}(2.4, 2.933)$	$U_{42}(3.2, 4.8)$	$U_{43}(3.8, 4.2)$	\times	$U_{45}(1.92, 2.08)$
$U_{51}(3.2, 4.8)$	$U_{52}(5.96, 6.04)$	$U_{53}(1.92, 2.08)$	$U_{54}(1.92, 2.08)$	\times

The numbers at the arcs in Fig. 5 denote the mean time in hours for transporters to go along that arc. Here are the other data:

i	1	2	3	4	5
wv_i	4	4	4	4	1
r_i	4	4	4	4	15
λ_i	3	3	3	3	3
g_i	15	15	15	15	15

p_i	0.1	0.1	0.1	0.1	0.6
-------	-----	-----	-----	-----	-----

As in Example 2.1 the inter-arrival times are assumed to be exponentially distributed and we assume $\lambda_i = 3/\text{day}$ for all locations. The using times UT are also exponentially distributed with $E(UT) = 2$ days. For the transportation costs we took the set-up case with set-up cost 50 per transporter and 2 per hour and per transported vehicle. The capacity of the transporters was set to 12 vehicles.

Both the case without reallocation and the above defined reallocation policies were investigated by LEO and FSAS. For the application of LEO we chose a population size of 100 chromosomes. The optimisation process stopped after 5 000 different chromosomes were investigated. The value of the goal function g from (2.1) was estimated for each chromosome by a simulation experiment of three runs with simulation time of 10 000 days. We remark that for our experiments the policies sS_1 and sS_n are implemented with the restriction $Q_i = q_i > 1$, $i = 1, 2, \dots, M$. This was done in order to limit the complexity of the underlying optimisation problem and to get passably reliable statements about the optimal parameter values for a given policy class and the corresponding criterion function in reasonable computing time. Thus in the case of five locations we have 16 parameters, one for the fleet size and five times the three parameters s_i , $Q_i = q_i$, and S_i for location i , instead of 21 parameters otherwise. Then, one optimisation process required approximately 8 hours on a 600 MHz Pentium PC. Altogether we realised three different optimisation processes for each policy by the choice of different starting values for the random number generator in LEO.

Table 1 contains the results for each of the three optimisation processes. The table's content must be interpreted in the following way: For the policies sQ_1 , sQ_n , SQ_1 , and SQ_n the first five policy parameters stand for the first parameter in the locations, and the second five for the second parameter. For instance, for policy sQ_1 the vector (53, 5, 4, 5, 6, 1, 6, 6, 7, 7, 20) means $K = 53$, $s_1 = 5$, \dots , $s_5 = 1$, $Q_1 = 6$, \dots , $Q_5 = 20$. For the policies sS_1 and sS_n the parameters are grouped in pairs s_i / S_i and the values for q_i , $i = 1, \dots, 5$.

Obviously, the results for policy π_0 – no reallocations at all– differ considerably from the results for each of the introduced reallocation policies. These really large differences are related to the optimal fleet size as well as to the optimal value of the criterion function. They may be typical for the investigated Example 2.1. In that example we have one particular surplus location (location 5) to which used vehicles will be returned with high probability. Thus, if an additional vehicle will be added to a sufficiently large fleet then the only consequence will be that on average one additional vehicle will wait in location 5. However if reallocations are allowed, an additional vehicle may produce a considerable increase of the overall gain. To summarise the comparison between π_0 and the implemented reallocation policies for the considered example we can say:

1. Wise reallocation policies improve the returns in orders of magnitude.
2. The optimal fleet size as well as the optimal gain for policy π_0 can be a bad approximation for the solution of the FSAP.

Let us now consider the results of our simulation optimisation experiments for the various reallocation policies. From Table 1 one can see that policy Sq_n is the favourite. On the second place is policy Sq_1 . This may indicate that surplus-based policies outperform the others.

Policy	K	Policy parameters										g
π_0	19	–	–	–	–	–	–	–	–	–	–	97.18
sQ_1	53	5	4	5	6	1	6	6	7	7	20	235.40
	53	8	6	7	7	1	6	6	7	7	11	231.25
	52	11	9	11	11	1	7	7	7	7	26	222.46
sQ_n	52	8	7	9	9	1	7	6	8	8	27	227.75
	53	9	7	9	10	1	8	6	7	7	40	225.26
	51	20	15	22	21	1	7	7	7	7	20	213.02
Sq_1	54	10	11	10	10	9	11	2	4	6	8	238.50
	57	12	12	12	12	11	12	2	6	11	9	232.50
	54	10	11	10	11	9	10	12	6	7	8	238.45
Sq_n	50	7	17	8	8	10	3	15	2	3	9	241.34
	51	8	13	8	8	12	3	4	2	2	11	241.26
	52	8	15	8	8	12	4	8	2	2	11	241.18
sS_1	51	5 / 24	5 / 20	6 / 14	5 / 24	1 / 24	6	6	6	7	11	236.17
	53	6 / 22	5 / 27	6 / 26	6 / 24	1 / 22	6	6	7	7	6	235.56
	53	6 / 22	5 / 20	5 / 22	6 / 28	1 / 22	6	6	7	7	7	235.28
sS_n	51	8 / 24	7 / 23	8 / 22	9 / 27	2 / 23	6	6	7	6	10	226.88
	49	7 / 24	6 / 13	8 / 23	8 / 21	1 / 21	5	5	8	7	8	226.51
	47	7 / 27	5 / 26	8 / 26	7 / 20	1 / 19	6	4	7	7	5	225.79

Table 1. The simulation optimisation results for the FSAP with five locations.

However, the reason for this we see in the concrete FSAP (Example 2.1), where location 5 is a surplus station, and all others are shortage stations. The parameter values found for the “optimal” Sq_n policy, given in Table 1, indicate that from the locations 1 to 4 do de facto not go any transhipments to other locations. The returning probabilities to these locations are so small that the levels $S_i \geq 7$, $i = 1, \dots, 4$, will be exceeded rarely. On the other hand location 5 immediately offers 9 vehicles as soon as there are more than 10 vehicles waiting (in the first optimisation process). Since the returning probability to location 5 is very high ($p_5 = 0.6$) this level will be exceeded after a short time. Consequently, transhipments will go on the whole out of location 5 to all other locations. Since it is easier to distribute 9 offered vehicles among 4 locations than to find a single receiver, policy Sq_n is slightly better than policy Sq_1 . As a summary to this argumentation we formulate

Hypothesis 1: For the FSAP a uniform best reallocation policy class does not exist.

Next, we will interpret the data of Table 1 with respect to the goal function g . The behaviour of function g heavily depends on the considered class of reallocation policies. Whereas for the classes Sq_n , Ss_1 and Ss_n partially very different fleet sizes and policy parameter values cause similar goal function values, we have a different situation e.g. for policy class sQ_1 . From this and from the complexity of the FSAP we conclude

Hypothesis 2: The goal function possesses a multitude of local optima.

Finally, the results of the optimisation process allow us saying some words about the quality of the used Genetic Algorithm. In almost all experiments the Genetic Algorithm “recognized” that the locations 1 to 4 are identical in a definite sense. Therefore, in the optimal solution of the FSAP the policy parameter values should be more or less identical for these locations. Most clearly this can be seen in Table 1 for policy classes sQ_1 and Sq_1 . However, we have to remark that the search process of the Genetic Algorithm was not yet finished. For instance, the solution of the first optimisation process for policy class sQ_1 was found as the 4 789th

considered chromosome (from 5 000 altogether). This indicates that further improvements of the goal function are possible if more computing time will be invested.

5. Outlook

In the paper we presented an approach to the FSAP, which seems to be very promising. Because we combined simulation with Genetic Algorithms it is possible to investigate fleet sizing and allocation problems of in principle arbitrary complexity. Some problems to be solved next are the following:

To confirm the hypotheses stated above by more examples.

To extend the simulator to more realistic transportation systems (other distributions, finite number of transporters, non-homogeneous vehicles, buy-and-sell actions, ...).

To design and to investigate other reallocation policy classes.

To improve the optimisation process (adjustment of the Genetic Algorithm to the FSAP).

To extend the optimisation to further parameters like, e.g., the queueing capacity for waiting clients.

Faster computation by distributed or parallel processing.

Finally, we thank M. Bogataj and an anonymous referee for their helpful comments on a first version of this paper.

References

1. Arnold, J.; Köchel, P. (1996). Evolutionary Optimization of a Multi-location Inventory Model with Lateral Transhipments. *9th Intern. Work. Seminar on Production Economics*, Igl, Pre-Prints v.2, 401-412
2. Beaujon, G.J.; Turnquist, M.A. (1991). A Model for Fleet Sizing and Vehicle Allocation. *Transportation Sci.*, 25, 19-45
3. Bolch, G.; Greiner, St.; de Meer, H.; Trivedi, K.S. (1998). Queueing Networks and Markov Chains. *John Wiley & Sons, Inc.*
4. Buzacott, J.A.; Shanthikumar, J.G.; Yao, D.D. (1994). Jackson Network Models of Manufacturing Systems. In D.D. Yao (ed.): *Stochastic Modeling and Analysis of Manufacturing Systems*. Springer-Verlag, 1994, 1-45
5. Cheung, R.K.; Powell, W.B. (1996). An Algorithm for Multistage Dynamic Networks with Random Arc Capacities, with an Application to Dynamic Fleet Management. *Oper. Res.*, 44, 951-963
6. Crainic, T.G. (2000). Service network design in freight transportation. *European Journal of Operational Research*, 122, 272-288
7. Du, Y.; Hall, R. (1997). Fleet Sizing and Empty Equipment Redistribution for Center-terminal Transportation Networks. *Man. Sci.*, 43, 1451-57
8. Fu, M.C. (1994). Optimization via simulation: a review. *Ann. of Oper. Res.*, 42, 199-247
9. Hader, S. (1998). A Multiagent simulation optimization system. In: Bargiela, A., Kerckhoffs, E. (eds.): *Simulation Technology: Science and Art. Proceed. of the 10th European Simulation Symposium*. Society for Computer Simulation International, San Diego, 1998, 124-128
10. Koenigsberg, E.; Lam, R.C. (1976). Cyclic Queue Models of Fleet Operations. *Oper. Res.*, 24, 516-529
11. Köchel, P. (1997). Finite Queueing Systems with Job Admission Control. *MMB '97*, 9. ITG/GI-Fachtagung, Freiberg, 1997, S. 137-148.
12. Köchel, P. (1997). An Approximate Solution for the Fleet Sizing and Allocation Problem.

- Promet-Traffic-Traffico*, 9, 195-200
13. Kunze, S. (2000). Optimisation of a system of removable resources through simulation and Genetic algorithms. *Diploma Thesis*, Technical University Chemnitz (in German)
 14. Nieländer, U. (1999). Simulation Optimization of Kanban Systems using a non-standard Genetic Algorithm. Paper presented at the 4th ISIR Research Summer School on Inventory Modelling 24-28 August 1999, Exeter, UK
 15. Nieländer, U. (1996). On the Optimal Configuration and Control of Discrete Systems, especially Manufacturing Systems, by Evolutionary Algorithms and Simulation. *Diploma Thesis*, Technical University Chemnitz (in German)
 16. Powell, W.B.; Carvalho, T.A. (1998). Dynamic Control of Logistics Queueing Networks for Large-Scale Fleet Management. *Transportation Sci.*, 32, 90-109
 17. Shantikumar, J.G.; Yao, D.D. (1987). Optimal Server Allocation in a System of Multi-Server Stations. *Man. Sci.*, 33, 1173-1180
 18. Wu, P.; Hartman, J.C.; Wilson, G.R. (1999). Fleet Sizing in the Truck Rental Industry. *Working Report No. 98W-009*, Lehigh University