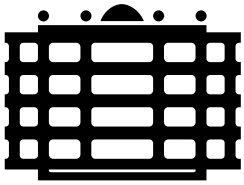


TECHNISCHE UNIVERSITÄT
CHEMNITZ

Fakultät für Informatik

CSR-09-03

Simulation Optimisation: Approaches, Examples, and Experiences

Peter Köchel

März 2009

Chemnitzer Informatik-Berichte

Simulation Optimisation: Approaches, Examples, and Experiences

Peter Köchel
Fakultät für Informatik
TU Chemnitz
e-mail: pko@informatik.tu-chemnitz.de

Abstract
Simulation based optimisation or simulation optimisation is an important field in stochastic optimisation. The present report introduces into that problem area. We distinguish between the non-recursive and recursive approaches of simulation optimisation. For the non-recursive approach we consider three methods, the retrospective, SPO-, and the RS-methods. With the help of a simple inventory problem we discuss the advantages and disadvantages of these methods. As a recursive method we consider in the second part of our report the coupling of simulation with Genetic Algorithms. As an application example we take a complex multi-location inventory model with lateral transshipments. From our experiences with such optimisation problems we finally formulate some principles, which may be relevant in simulation optimisation.

1. Introduction

Simulation is an experimental method, usually aided by computer technique, to analyse systems, which may be not suited for real experiments or whose corresponding models may be too complex for analytical investigations. During the last decade simulation became a more and more important tool for performance analysis, design and control of complex systems. In general, simulation estimates relevant for the problem performance measures by corresponding experiments. However, such estimation does not solve any optimisation problem. To realise an optimisation in a stochastic environment the simulation based estimation must be combined with an optimisation to the ***Simulation Optimisation*** approach (SO approach). The application of simulation optimisation to problems of the optimal design and/or control of systems, however, at present time is not common. On the other side, more and more actual optimisation problems are not tractable for classical approaches available e.g. in operations research. Typical for such problems is that no analytical form of the criterion(s) is available, that the problems have multiple optima and no well-structured solution space. Reasons for that may be complicated stochastic influences, highly inter-dependent elements of the system, complex performance measures and others. In many of such cases interesting performance measures and the resulting criterion function for a given system design as a rule can be estimated only by simulation. This leads to some complications. For instance, it is no more possible to calculate a gradient for the goal function. As well it is problematic to compare two criterion function values by their estimations.

In the past a great variety of approaches for SO is developed. Figure 1 contains a classification of SO approaches, which is given by TEKIN & SABUNCUOGLU (1994), based on a rough grouping of optimisation problems. Comprehensive reviews are presented by FU (1994), FU et al. (2005), TEKIN & SABUNCUOGLU (2004), and others. A textbook with a strong Mathematical orientation is PFLUG (1996). Applications of the SO approach exist in many fields. With respect to operations research problems we mention above all inventory models (KÖCHEL & NIELÄNDER, 2005), logistic systems (KÖCHEL et al.), and manufacturing systems (KÖCHEL & NIELÄNDER, 2002; KÄMPF & KÖCHEL, 2006).

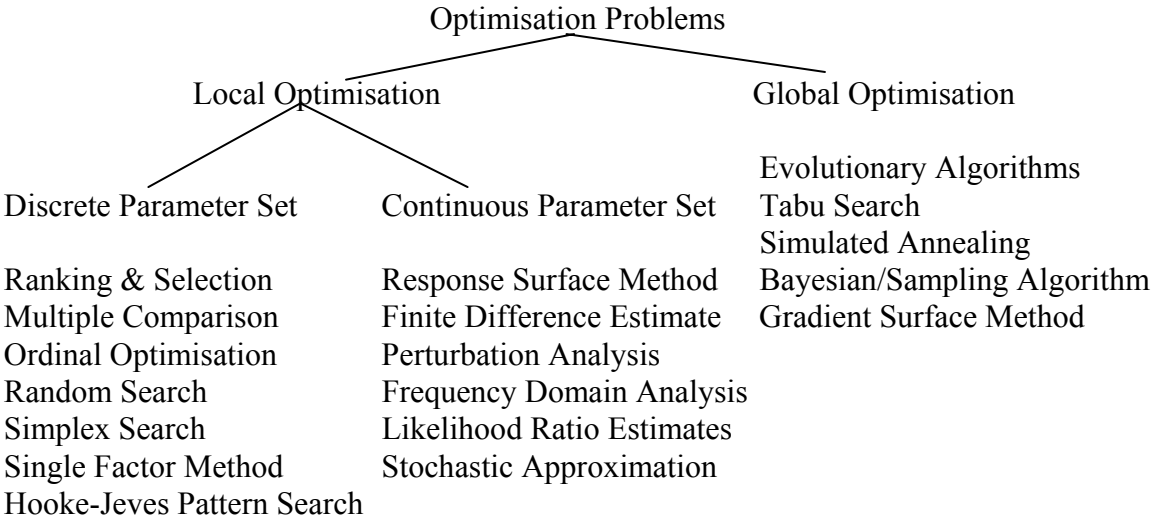


Figure 1. Grouping of SO-approaches by Tekin & Sabuncuoglu (2004)

At Chemnitz University of Technology, chair of Modelling and Simulation, the starting point for the research in the field of SO dates back to 1995. The necessity to deal with such problems was formed in connection with the development and improvement of the knowledge-based optimisation system DIM_EXPERTE (see e.g. HADER, 1998), which was designed for the optimisation of complex control problems above all from such fields like inventory, logistics, or manufacturing systems. At present time at the chair of Modelling and Simulation are developed several software tools for SO. We mention only the systems LEO (NIELÄNDER, 1999), PATHOS (Hader, 2001), KASIMIR (Köchel et al., 2001), CHAOS (KÄMPF, 2008), and CHEOPS (NIELÄNDER, 2009). In addition to that we can report on various applications of the SO approach to complex optimisation problems in inventory theory, logistics, and manufacturing systems (ARNOLD et al., 1996, 1997, 1998; HOCHMUTH & KÖCHEL, 2009; KÄMPF, 2008; KÄMPF & KÖCHEL, 2006; KÖCHEL, 1998b; KÖCHEL et al., 2003; KÖCHEL & NIELÄNDER, 2002, 2005). Therefore it is time to review and summarise what was done in the past. A second aim of the present report is that we want to give the students, which are visiting the lectures on discrete simulation or on evolutionary optimisation some additional material on an important and actual research field on the interface between computer science, operations research, and practical applications.

We divide the present report into two parts. In the first part we concentrate on the non-recursive approach of simulation optimisation. Chapter 2 starts with a brief introduction into the class of optimisation problems we are considering. We define also important basic notions, which are illustrated by some relevant for practice examples. Furthermore, we characterise two possible approaches for simulation optimisation - the non-recursive and the recursive approaches. In Chapter 3, after the principal description of the non-recursive approach, we consider three methods in more detail: the retrospective approach, the SEO-approach, and the Response Surface methodology. We demonstrate their working principles by a simple inventory problem and discuss the pros and cons.

Part two is dealing with the recursive approach. Thereby we concentrate on the coupling of simulation and Genetic Algorithms. After a brief discussion of the recursive principle in Chapter 4 we demonstrate in Chapter 5 our procedure by a complex problem from inventory theory. Furthermore in Chapter 6, we briefly discuss the question which elements and features a simulator should have in order to be suited for simulation optimisation e.g. in inventory theory. Some remarks on our own experiences with simulation optimisation of different stochastic models conclude the report.

2. Optimisation versus Simulation

When we speak about optimisation in connection with systems we mean above all two things: ***Design*** of optimal in a defined sense systems and optimal ***control*** of systems. In both cases we have to solve optimisation ***problems*** (OP) of the following kind

$$g(\theta) \quad \rightarrow \quad \min_{\theta \in \Theta} \tag{2.1}$$

with explicitly known ***criterion function*** g and ***decision set*** Θ. We abbreviate such problems by OP(g, Θ), where Θ denotes the set of all ***admissible decisions***. The corresponding criterion function is assumed to be a real valued function g: $\Theta \rightarrow R$. Value $g(\theta)$ represents the cost connected with the admissible decision $\theta \in \Theta$.
With several regularity assumptions on g and Θ the ***set of optimal decisions***

$$\Theta^* := \{ \theta^* \in \Theta : g(\theta^*) = \inf [g(\theta): \theta \in \Theta] \}$$

is not empty. The value $g(\theta^*)$ is called ***optimal value***. A decision $\theta' \in \Theta$ is a ***locally optimal decision*** if there exists an environment U of θ' such that

$$g(\theta') = \inf\{ g(\theta): \theta \in \Theta \cap U \}.$$

An OP(g, Θ) is solved if we have found an optimal decision θ^* and the optimal value $g^*=g(\theta^*)$. The pair (g^*, θ^*) is called a ***solution*** of OP(g, Θ).

Example 2.1: $g(\theta) = c / \theta + d \cdot \lambda\theta / (1 - \lambda\theta)$ with c, d, $\lambda > 0$ and $\Theta = \{\theta: \theta \in (0; 1)\}$.

Obviously, g is a convex function with the unique optimal decision

$$\theta^* = \frac{\sqrt{c}}{\left(\lambda \cdot \sqrt{c} + \sqrt{\lambda \cdot d}\right)} \in (0; 1).$$

As optimal value we calculate $g(\theta^*) = \lambda \cdot c + 2 \cdot \sqrt{\lambda \cdot c \cdot d}$.

An OP as in Example 2.1 can originate from the optimisation of a ***stochastic system***. For this we consider the following example.

Example 2.2: M/M/1/∞ - queueing system

In a service system arrive jobs with an arrival rate of λ job per hour. These jobs are served in accordance with the FIFO-discipline (First-In-First-Out) with a service rate of μ jobs per hour. We assume infinite many waiting places and that it is possible to chose the service rate in the interval $\lambda < \mu < \infty$ in such a way that there exists steady-state regime. Following cost arise in the system:

a) Service cost proportional to the service rate, i.e., $c \cdot \mu$ monetary units per hour, $c > 0$.
b) Sojourn cost of $d > 0$ monetary units per hour and job staying in the system.

Because of the number of jobs staying in the system is a random variable, depending on the service rate μ, we consider the expectation $L(\mu)$ of that number in the steady-state regime. Thus function $g(\mu)= c \cdot \mu + d \cdot L(\mu)$ describes for the steady-state regime the average cost per hour under service rate μ. For the considered service system it holds that $L(\mu) = \lambda / (\mu - \lambda)$. If we replace the service rate by the expected service time $\theta = 1 / \mu$ then we get the criterion function from Example 2.1. In other words we get the result that the service rate $\mu^*= 1/\theta^*$ with $\theta^* = \sqrt{c} / (\lambda \cdot \sqrt{c} + \sqrt{\lambda \cdot d})$ minimises the average cost per hour in the steady-state. These cost are equal to $g(\theta^*) = \lambda \cdot c + 2 \sqrt{\lambda \cdot c \cdot d}$ monetary units per hour.

Example 2.3: (S, S) - inventory problem

At the beginning of a given time period a single store has to define for a given product such an inventory level that minimises the total cost. The demand during the period is assumed to be random and can be described by a non-negative random variable X with distribution function $F(\cdot)$ and density function $f(\cdot)$. We assume following cost factors:

- Inventory cost factor $h > 0$ for each not demanded product unit at the end of the period.
- Shortage cost factor $p > 0$ for each unsatisfied demand unit at the end of the period.

Let θ_{max} denote the capacity of the store and θ the inventory level chosen at the beginning of the period. Then $\Theta = \{ \theta: 0 \le \theta \le \theta_{max} \}$ denotes the set of admissible inventory decisions. Let further $g(\theta)$ denote the expected single-period cost for given $\theta \in \Theta$. Obviously it holds that

$$g(\theta) = \int_0^{\theta} h \cdot (\theta - x) f(x)\, dx + \int_{\theta}^{\infty} p \cdot (x - \theta) f(x) dx. \tag{2.2}$$

Again we have a convex criterion function. The unique minimum point θ' of function g can be calculated from $dg(\theta)/d\theta = 0$ as solution of the equation $F(\theta') = p / (h + p)$. Subject to the capacity of the store θ_{max} and the convexity of the criterion function we get the optimal decision for the considered inventory problem as $\theta^* = \min(\theta', \theta_{max})$, i.e., a so-called (S, S) ordering or order-up-to policy with $S^* = \theta^*$ is optimal. Such policies are well-known in inventory theory and easy to realise. An (S, S) policy orders up to inventory level S in case the starting inventory is below the level S. Otherwise nothing is ordered.

Generalisation of the Examples 2.2 and 2.3 leads to the following formalisation. For a decision $\theta \in \Theta$ we describe the quality of a stochastic system through a ***quality function*** $B(\theta, X(\theta))$, where random variable $X(\theta)$ with distribution $F_\theta(\cdot)$ represents the random input of the system respectively the stochastic influences on the system. For a service system for instance quality function $B(\theta, X(\theta))$ can represent the number of jobs in the system in steady-state regime, $F_\theta(\cdot)$ the distribution function of the service time, and θ the average service time. Since the quality function B itself is a random variable and because of the comparison of random variables is not so easy mostly the expected value of the quality function is used as ***performance measure*** for a stochastic system. This means that we consider for given decision $\theta \in \Theta$ the expected quality in the steady-state regime, i.e.,

$$g(\theta) = E [\ B(\theta, X(\theta))\ ]. \tag{2.3}$$

Thus we get a ***stochastic optimisation problem***

$$g(\theta) = E [\ B(\theta, X(\theta))\ ] \quad \underset{\theta \in \Theta}{\rightarrow} \quad \text{Min.} \tag{2.4}$$

By taking the expected value of the systems performance as assessment basis we reduce optimisation problems for stochastic systems to the form of (2.1). Despite of the unique formal representation we distinguish between stochastic and deterministic optimisation problems. Typical for a stochastic OP is:

1. Not all data relevant for the solution are available at the moment the decision must be taken. For instance the managing clerk in a store does not know the future demand at the beginning of a period.
2. Often stochastic influences on the system depend on the chosen decision. In Example 2.2 the distribution function of the service time depends on the service intensity.

The most essential consequence of these typical properties of a stochastic OP is that a solution is much more complicated as for a deterministic OP. Thus we need for stochastic OP own special solution methods. Basis of any optimisation is the analysis or valuation of a given decision respectively of a given real configuration $\theta \in \Theta$ of the system, that is the ability to calculate the value $g(\theta)$ of the criterion function. In the Examples 2.1 to 2.3 this could be done without any problems. Moreover, in these examples it was possible to use the first derivative of the criterion function and to get a closed-form representation of the corresponding optimal decision and partially of the optimal value also. This is more or less the ideal case of an OP. For the following example however this is no more possible in general.

Example 2.4: Linear transportation problem

A single product, manufactured at m locations, must be transported to n consumer location in such a way that the total cost will be minimised. Let a_i denote the amount of product manufactured at location i, b_j the demand at location j, θ_{ij} the amount of transhipped product from i to j, and c_{ij} the transportation cost per transhipped product unit, $i = 1..m, j = 1..n$. Then the corresponding (deterministic) OP is characterised by the criterion function

$$g(\theta) = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij}\ \theta_{ij} \quad \underset{\theta \in \Theta}{\rightarrow} \quad \text{Min}$$

and $\Theta = \{\theta = (\theta_{ij}) :\ \theta_{ij} \geq 0, i = 1..m, j = 1..n; \quad \sum_{i=1}^{n} \theta_{ij} \leq a_i, i=1..m; \ \sum_{j=1}^{m} \theta_{ij} \geq b_j, j = 1..n\}$.

Decision set Θ here is given implicitly through a finite number of inequalities. A (transhipment-) decision is represented as a (transshipment-) matrix with m rows, the delivery locations, and n columns, the receiver locations.

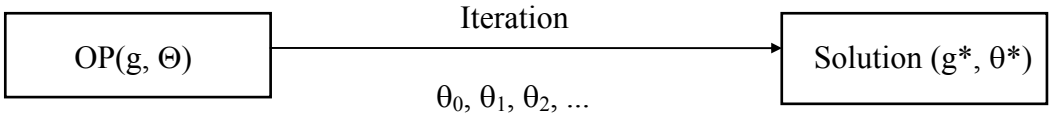


Figure 2. Scheme of the iterative solution process

To solve such problems various ***iterative algorithms*** exist (see their scheme in Figure 2). Thus for some optimisation problems solutions exist in a closed form (Examples 2.1 to 2.3) whereas for other problems (as in Example 2.4) a solution is possible only through the application of an iterative algorithm. However, for complex problem formulations it is not possible to apply any of these possibilities because of the values $g(\theta)$ of the criterion function for $\theta \in \Theta$ cannot be calculated or at most only with an (possibly unknown) error. An example for this is the following inventory model from KÖCHEL (1982, 1998).

Example 2.5: Multi-Location-Model with Transshipments (MLMT)

For $M \geq 2$ locations we have to organise an optimal inventory management under following assumptions:

- Increase of the inventory in each location through an ordering decision at the beginning of a period.
- Satisfaction of the random demand in each location during the period.
- Redistribution of the surplus inventory at the end of the period through a transshipment decision.

The verbal formulation of the corresponding OP is: Define such ordering and transshipment decisions, which minimise the expected total cost of the inventory system.

The criterion function for the OP from Example 2.5 has no analytically tractable form. The reason for that is that at the end of a period we have to solve a linear transportation problem, which delivery and receiver locations as well as the available product depends on the remaining not used up to now product in all *M* locations. For this, however, only an iterative algorithm exists, which must be applied for the calculation of the value of the criterion function for any possible demand realisation. This is not possible for continuous demand with infinite many realisations. One alternative in such situations is to replace the initial system by another one, which is more easily to analyse, respectively to solve an approximate problem $OP(g_N, \Theta_N)$ instead of problem $OP(g, \Theta)$. The result of that approach is an approximate solution (g_N^*, θ_N^*). The scheme of that approach is shown in Figure 3.

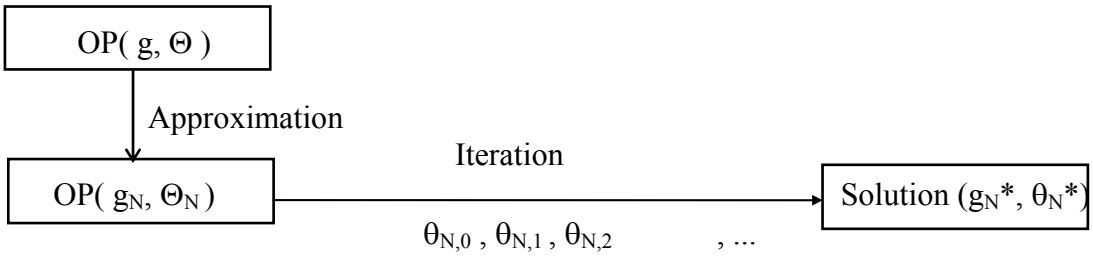


Figure 3. Scheme of the iterative solution process under use of an approximate problem

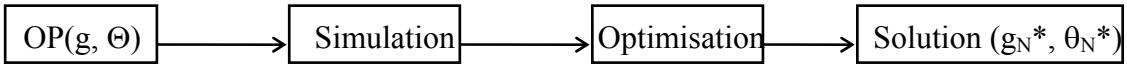
Among all possibilities to replace the initial problem by an approximation $OP(g_N, \Theta_N)$ at present time simulation is the broadest applicable and most successful one. In case that an adequate to the problem simulation model is used the assessment of a given decision $\theta \in \Theta$ as the rule is realised in two steps.

Step ***Simulation***: Generate *N* realisations $X_1(\theta)$ to $X_N(\theta)$ of the relevant random variables.

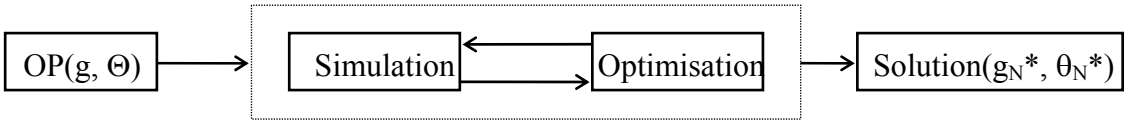
Step ***Estimation***: Use the sample average $g_N(\theta) = 1/N \sum_{i=1}^{N} B(\theta, X_i(\theta))$ as estimation for $g(\theta)$.

Obviously, in such a way simulation alone does not solve any optimisation problem. But we can combine simulation with corresponding optimisation methods to a tool, which allows a **simulation-based optimisation**. In principle exist two approaches to realise the combination of simulation and optimisation - the **recursive** and the **non-recursive** approaches (see FU, 1994; FU & HEALY, 1997; PFLUG, 1996). In the non-recursive approach simulation and optimisation are decoupled in such a way that two separate problems must be solved – a simulation and estimation problem and an optimisation problem. The idea behind that separation is simple: In a first step generate realisations of those random variables, which influence the behaviour of the system (simulation), and accept these realisations as real or use them to approximate the original criterion function (estimation). The resulting deterministic optimisation problem must be solved in the concluding second step (optimisation). An optimisation in this approach is realised on the basis of the observed or simulated behaviour of the stochastic system. HEALY & SCHRUBEN (1991) call this approach also the ***retrospective*** approach. In opposition to this we have the ***prospective*** or ***recursive*** methods, in which the simulation and optimisation steps alternate with each other (see Fig.4). In difference to the non-recursive approach we have here no complete decoupling. For any new decision, suggested by the optimiser, information about the decisions performance is collected

through simulation. Optimisation and simulation push each other as long as we have not found an acceptable decision.



a) The non-recursive approach



b) The recursive approach

Figure 4. Schemes of non-recursive and recursive simulation-based optimisation

In the following chapters we discuss both approaches in more detail. We restrict our considerations to stochastic optimisation problems of the form (2.4), where the quality function B itself is a random variable or can be calculated with a random error only. For better understanding we define as ***simulation-based optimisation problem*** an optimisation problem for which the criterion function represents the performance established through simulation. With this notion the answer to the question „Optimisation versus simulation?“ is „Optimisation via simulation!“ respectively simulation optimisation. Besides a considerable enlargement of now solvable optimisation problems however arise some new questions. One of the most important is how to find the optimum if we can measure the criterion function only with a random error. Figure 5 illustrates the problem. On the first glance we see two errors comparing the corresponding optimal decisions (g*, θ*) and (g_N*,θ_N*).

1. The optimal decisions θ* and θ_N* are very different.
2. The optimal values g* and g_N* are also different.

The second error is not so fatal. More important for practice is the difference between g(θ*) and g(θ_N*), which in case of a picture as in Fig.5 can be much more greater than between g* and g_N*. It is not possible to eliminate such errors if we chose decisions on the basis of (random) experiments. Consequently, we should have this in mind that inferences based on a simulation-based optimisation always have their limits.

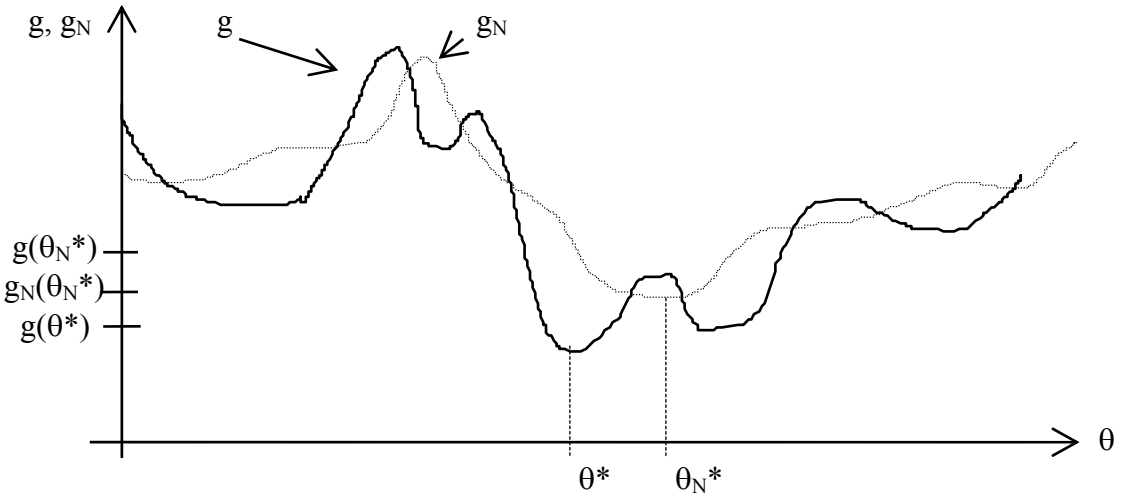


Figure 5. Sketch of a possible behaviour of the criterion functions g and g_N

3. Non-recursive approaches in simulation optimisation

We start with a more detailed consideration of the non-recursive approach. For that reason let us look at the scheme in Fig.6. It shows how the consecutive application of a ***simulator*** and an ***optimiser*** generates an approximate solution (g_N^*, θ_N^*) for a given OP(g, Θ). In a first stage the simulator generates a sample $X_1, X_2, \ldots, X_N$ of size *N* for relevant random variables. The task for the optimiser in the second stage is to find an optimal decision only on base of the generated sample. For that the optimiser should involve at least four elements (see Fig.6) A ***starter*** chooses an initial decision $\theta \in \Theta$, which will be over handed to the ***assessor***. Using the sample values for the random variable (quality function) $B(\theta, X(\theta))$ the assessor calculates an estimation $g_N(\theta)$ for the true value $g(\theta)$ of the criterion function. On basis of that estimation (and may be some additional available information e.g. on former considered decisions and/or structural properties of the optimisation problem) a ***decider*** else accepts the actual investigated decision as approximately optimal or a ***searcher*** gets the order to find another decision, which hopefully is an improvement with respect to all up to now considered decisions. These four elements of the optimiser should be concretised in accordance to the applied approach. In the following we will consider some of such approaches.

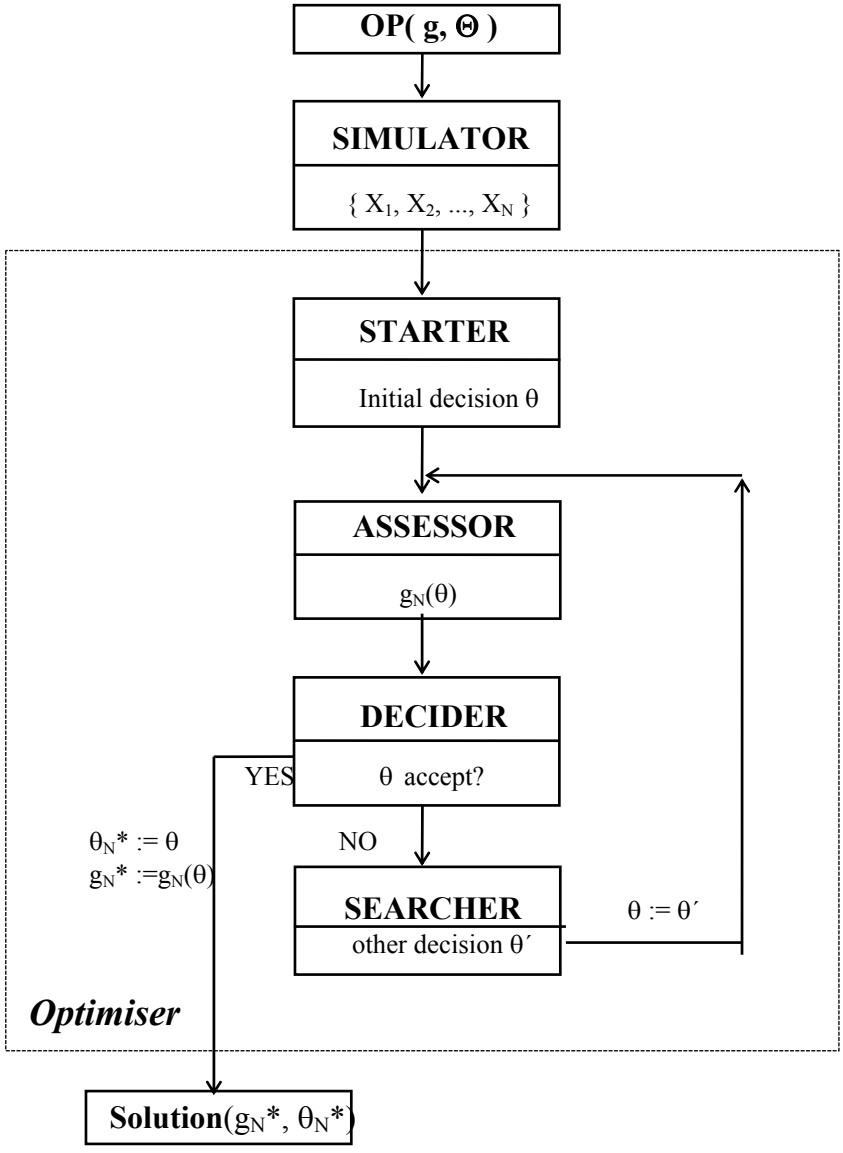


Figure 6. Scheme of simulation-based non-recursive optimisation

3.1. The retrospective approach

The basic idea of the retrospective approach suggests to generate realisations of relevant for the corresponding system random variables (Simulation) and to define for the resulting realisations optimal parameter values $\theta \in \Theta$ (Optimisation). Thereby it plays no role in which form we realise the optimisation process. The application of the retrospective approach postulates the condition that the distribution function $F(\cdot)$ of the random system input X is independent on $\theta \in \Theta$. In such cases we can describe the retrospective approach for the solution of OP(g, Θ), assuming that the initial criterion function g is replaced by the sample average g_N of approximated respectively estimated performance values, in the following way:

1. Distribution function F of random variable X is approximated by the empirical distribution function F_N. We get an OP(g_N, Θ) with
$$g_N(\theta) = \int B(\theta, x) dF_N(x) = \frac{1}{N} \sum_{i=1}^{N} B(\theta, X_i), \qquad (3.1)$$
where X_1, X_2, ..., X_N are independent and identical as X distributed random variables.
2. We solve problem OP(g_N, Θ), i.e., $g_N \rightarrow \underset{\theta \in \Theta}{\text{Min}}$.

For better understanding how works that approach and which questions must be solved applying it we consider its application to the (S, S) inventory problem from Example 2.3. From Example 2.3 we have that the quality function for any realisation *x* of the demand has the following form:

$$B(\theta, x) = h \cdot (\theta - x)^+ + p \cdot (x - \theta)^+ = \begin{cases} h \cdot (\theta - x), & 0 \le x \le \theta; \\ p \cdot (x - \theta), & \theta < x. \end{cases} \qquad (3.2)$$

We use $z^+ = \max(0; z)$ for real z. Decision θ denotes the inventory at the beginning of a period. In the first step, the simulation step (cp. Fig.4), we simulate *N* independent exemplars X_1 to X_N of the demand X. Let x_1 to x_N denote the observed values (realisations), and let denote $\mu_N = 1/N\ (x_1 + x_2 + \ldots + x_N)$ the corresponding sample average of the demand. The ordered sequence of sample elements (demand values) we denote by $\{x_{(1)}, x_{(2)}, \ldots, x_{(N)}\}$. If we assume $x_{(0)} = 0$ and $x_{(N+1)} = \infty$ then it holds that $0 = x_{(0)} < x_{(1)} \le x_{(2} \le \ldots \le x_{(N)} < x_{(N+1)} = \infty$. From (3.1) and (3.2) follows immediately that

$$g_N(\theta) = p \cdot (\mu_N - \theta) + \frac{h + p}{N} \sum_{n=1}^{k} (\theta - x_{(n)}) \ \textit{for}\ \theta \in [x_{(k)}, x_{(k+1)}), k = 0..N. \qquad (3.3)$$

To develop an algorithm, which defines that optimal inventory level θ_N^* for which the criterion function g_N from (3.3) will be minimised, we need some preliminary analytical investigations. From (3.3) follows that g_N is a continuous and peace-wise linear function with respect to θ. Consequently the minimum will be attended at one of a border point of one of the intervals $I(k) := [x_{(k)}, x_{(k+1)})$, $k = 0..N$. However, that border point can not be the point $x_{(0)} = 0$ because of from (3.3) follows that function $g_N(\theta) = p \cdot (\mu_N - \theta)$ is decreasing for $\theta \in I(0) \cup \{x_{(1)}\}$. In the same way we can exclude point $x_{(N+1)}$ because of again from (3.3) follows that g_N is increasing in θ. Therefore we must check all observed demand values $x_{(1)}$ to $x_{(N)}$ for optimality. For small sample sizes, i.e. small *N*, it is no problem to consider all demand values. For big *N* more efficient algorithms are needed. For this reason another property of the criterion function g_N is helpful. From (3.2) it obviously follows that function $B(\cdot, x)$ is convex in θ for any fixed *x*. With $B(\cdot, x)$ also g_N is convex in θ. Therefore we have a simple

method to minimise g_N: We are searching the first interval I(k) in which function g_N is increasing in θ. The left border point of that interval is the optimal inventory level. Thus we are looking for an interval I(k) with the property $g_N(x_{(k)}) < g_N(x_{(k)} + \delta)$ for a $\delta > 0$. From (3.3) follows that this property is equivalent to the inequality p / (h + p) < k / N. At the end we have the following solution for the considered inventory problem:
a) The optimal inventory level is equal to

$$\theta_N^* = x_{(k^*)} \quad \text{with} \quad k^* = \min\{n:\ n / N > p / (h + p)\}. \tag{3.4}$$

b) For the optimal value holds

$$g_N(\theta_N^*) \quad = \quad p \cdot \left(\mu_N - x_{(k^*)} \right) + \frac{h+p}{N} \sum_{n=1}^{k^*} \left(x_{(k^*)} - x_{(n)} \right). \tag{3.5}$$

Remark 3.1
Interesting with respect to this solution is the fact that the optimal rank number k* does not depend on the sample values of the demand. The optimal inventory level naturally depends on the observed demand realisations. Equation (3.4) corresponds ideally with the similar formula from Example 2.3 when we remember ourselves that in our chosen retrospective approach the original distribution function F of demand X is replaced by the empirical distribution function F_N. Obviously, for the latter holds that $F_N(x) = k / N$ for $x \in I(k)$, $k = 0..N$.

To compare for the inventory problem the approximate solutions from (3.4) and (3.5) with the exact solution we consider now a concrete numerical example with following assumptions:
Cost [€]: h = 1 and p = 3.
Demand [item units]: X is exponentially distributed with parameter λ = 0.1, i.e.,

$$F(x) = 1 - e^{-0.1\,x} \text{ for } x \geq 0.$$

A) Exact solution.
For exponentially distributed demand X we get from (2.2) for inventory level θ ≥ 0 that

$$g(\theta) = h \cdot (\theta - \mu) + (h + p) \cdot \mu \cdot e^{-\theta / \mu}, \tag{3.6}$$

where μ = 1 / λ = E(X).
The optimal inventory level θ* is, using the equation F(θ*) = p / (h + p), equal to

$$\theta^* = - \mu \cdot \ln[h / (h + p)]. \tag{3.7}$$

If we put this expression into (3.6) we get the optimal value as

$$g(\theta^*) = h \cdot \theta^* = h \cdot \mu \cdot \ln[h / (h + p)]. \tag{3.8}$$

With the assumed data we finally deduce from (3.6) to (3.8) the concrete criterion function $g(\theta) = \theta - 10 + 40 \cdot e^{-0.1\,\theta}$ and the solution (g*, θ*) = (13.86; 13.86).

B) Approximate solution (retrospective approach)
Let us assume now that besides a sample of N observed or simulated demand values we have no further information on the demand. At first let us look at the development of the optimal rank number k* from (3.4) for different sample sizes. With the assumed cost parameters it follows from (3.4) that $k_N^* = \min\{n: n > 0.75 \cdot N\}$ for given sample size N. Thus we have:

N	1	2	3	4	5	6	7	8	9	10	20	50
k_N*	1	2	3	4	4	5	6	7	7	8	16	38

Table 1. Optimal rank numbers for the inventory model

From Table 1 we can immediately see for given sample size which number of the ordered sample defines the approximate decision θ_N*. It is interesting that for small sample sizes, here up to $N = 4$, the highest demand realisation defines the optimal inventory level θ*.
For further considerations we need now concrete demand realisations. Let us assume that we got values given in Table 2.

n	x_n	x_{10+n}	x_{20+n}	x_{30+n}	x_{40+n}	x_{50+n}	x_{60+n}	x_{70+n}	x_{80+n}	x_{90+n}
1	3.12	0.20	9.60	30.58	4.68	5.67	9.73	19.66	0.67	6.54
2	1.72	4.93	7.61	19.52	3.44	4.88	10.88	10.41	16.82	2.65
3	11.49	28.13	25.77	5.06	2.08	58.09	30.05	0.35	0.77	3.24
4	3.54	15.42	7.44	8.96	20.56	28.65	15.56	2.67	0.13	3.83
5	0.78	9.34	12.38	15.95	19.11	1.46	12.04	0.78	11.00	10.24
6	0.14	23.43	6.27	13.94	3.33	4.19	0.80	1.68	1.95	3.48
7	0.60	0.17	7.09	0.61	8.37	21.37	0.67	3.58	0.52	14.19
8	1.72	0.10	1.39	5.69	27.65	1.80	1.19	70.25	5.62	18.77
9	0.76	3.70	2.07	13.39	12.17	21.04	5.66	4.46	5.18	10.19
10	27.97	0.37	19.73	6.56	0.99	1.79	29.76	0.60	7.70	3.48

Table 2. Demand realisations for the inventory model

We calculate the approximate solutions for $N = 5, 10, 20, 30, 40, 50, 100$. We start with $N = 5$. From the first five values in Table 2 we calculate the corresponding sample average as $\mu_5 = 4.13$. The ordered sample is

$$x_{(1)} = 0.78, \quad x_{(2)} = 1.72, \quad x_{(3)} = 3.12, \quad x_{(4)} = 3.54, \quad x_{(5)} = 11.49\ . \tag{3.9}$$

With these values we get from Equ.(3.3) the (approximate-) criterion function g_5 as

$$g_5(\theta) = \begin{cases} 12.390 - 3.0\cdot\theta & \textit{for } 0.00 \le \theta \le 0.78; \\ 11.766 - 2.2\cdot\theta & \textit{for } 0.78 < \theta \le 1.72; \\ 10.390 - 1.4\cdot\theta & \textit{for } 1.72 < \theta \le 3.12; \\ 7.894 - 0.6\cdot\theta & \textit{for } 3.12 < \theta \le 3.54; \\ 5.062 + 0.2\cdot\theta & \textit{for } 3.54 < \theta \le 11.49; \\ -4.130 + 1.0\cdot\theta & \textit{for } 11.49 < \theta < \infty. \end{cases} \tag{3.10}$$

From Table 1, (3.9) and (3.10) follows immediately that $k^* = 4$, $\theta_5^* = 3.54$, and $g_5(\theta_5^*) = 5.77$. Comparing this solution with the exact solution $(g^*, \theta^*) = (13.86; 13.86)$ we see a strong underestimation of solution (g^*, θ^*) by the approximate solution. Moreover, from (3.6) follows $g(\theta_5^*) = g(3.54) = 18.23$, i.e., the "true" expected cost for inventory level $\theta_5^* = 3.54$ are about 50% higher than the minimum. Of cause that is a consequence of the small sample size. Only five sample elements is on one hand too less and on the other hand with the sample average $\mu_5 = 4.13$ they give a very inexact estimation of the true average demand $\mu = 10$. The results for the other N-values are given in Table 3 above the bold line.

N	5	10	20	30	40	50	100
θ_N*	3.54	3.54	11.49	11.49	12.38	13.39	13.39
$g_N(\theta_N$*)	5.77	11.30	14.41	13.24	11.63	13.17	13.39
$g(\theta_N$*)	18.32	18.23	14.17	14.17	13.98	13.87	13.87
μ_N	4.13	5.18	6.88	7.90	8.93	9.15	9.64
θ_N	5.73	7.18	9.54	10.95	12.38	12.68	13.36
$g(\theta_N)$	18.28	16.69	14.95	14.33	13.98	13.94	13.88

Table 3. Results for the inventory model with the true solution (g*, θ*) = (13.86; 13.86)

With increasing complexity of the optimisation problem the realisation of the just demonstrated retrospective approach will be more and more complicated. That fact show e.g. two other papers, which are dealing with inventory problems. A generalisation of Example 2.3, where fixed ordering costs are assumed, is considered in FU & Healy (1997). In this case a so-called (s, S) policy is optimal. Such a policy depends on two parameters. The most important difference to the problem from Example 2.3 is that the search for optimal parameter values can no more be restricted to the points of corresponding demand realisations. KÖCHEL (1998.b) has applied the retrospective approach to a MLMT (see Example 2.5). He developed a corresponding search algorithm to find optimal order decisions. However, it is not possible to transform this approach to a MLMT with more than two locations (see also Chapter 4).

Besides the just considered and demonstrated by an example approach, where the distribution function is approximated by the corresponding empirical distribution, there exist other non-recursive approaches. Let us briefly consider one, probably the simplest non-recursive approach – the *SEO* approach (*S*imulation – *E*stimation – *O*ptimisation), where unknown parameters of related random variables are replaced by their sample averages. Applying SEO to the (S, S)-inventory problem we estimate the expected demand μ_N, treat this value as the true one, put it into (3.7), and calculate the inventory level $\theta_N = -\mu_N \cdot \ln[h/(h+p)]$. Again from (3.6) we can calculate the corresponding expected cost $g(\theta_N)$. For the data from Table 2 we get for sample size $N = 20$ that $\theta_{20} = -6.88 \cdot \ln[0.25] = 9.54$, and from (3.6) follows $g(9.54) = 14.95$. In Table 3 one can find the corresponding results for the other N-values below the bold line.

Despite of the relatively good results of the *SEO*-approach for our inventory problem (the reason for that is the flat behaviour of the criterion function in the region nearby the minimum) we strongly recommend not applying it in general. The argument for that comes from JENSEN's inequality (e.g. PFLUG, 1996), which says the following:

For a convex function g and a random variable X with finite expectation E(X) holds

$$g(E(X)) \le E(g(X)).$$

The risk of gross false estimations is very big. Let for instance g be a function that is defined for x = 0, 1, 2 with values $g(0) = g(2) = 1$ and $g(1) = 0$. Let X be a random variable, which takes values 0, 1, 2 with equal probability 1/3. Then we have $E(X) = 1$, $g(E(X)) = g(1) = 0$, but $E(g(X)) = 1/3 \cdot (1 + 0 + 1) = 2/3$.

Which are the main conclusions from the above considered (S, S) inventory problem? We remember that the exact solution is (g*, θ*) = (13.86; 13.86).

1. The considered non-recursive approach is not suited for small sample sizes. For more complex problems a careful planning of the simulation experiment is necessary. The minimum number of the sample size N for statistically proved statements is heavily dependent on the problem and not easy to define.

2. The calculated solutions for $N = 20, 30, 40$ are relatively bad approximations for the true solution, but their expected cost are not so far from the minimal value. This is a consequence of the convexity of the performance function g and its flat behaviour in a relatively wide area around the optimum. But the proof of such structural properties in general may be not possible or only with considerable effort.
3. The quality of the estimators for parameters of random variables, which influence the behaviour of the system, have a great impact on the quality of the approximate solution. In our example the sample average of the demand is such a parameter.

These conclusions on the applicability of the considered non-recursive approach are supported by experiences in connection with other investigations (cp. for instance FU & HEALY, 1997; KÖCHEL, 1998). Thus we formulate the following ***summarising appraisal***:

1. The retrospective approach finds relatively fast a solution for simple problems.
2. To define a qualitatively good solution a high simulation effort is necessary.
3. The practical application makes sense only in the case that the relevant for the problem random variables do not depend on the chosen decisions and if the value of the criterion function for different decisions can be estimated without repetition of the whole simulation.
4. Without previous analytical investigations of the original problem the non-recursive approach as a rule is not practicable.

3.2. The Response-Surface-Methodology

A generalisation of the above-demonstrated approach is the *R*esponse *S*urface *M*ethodology (RSM). With RSM is connected a class of widespread and theoretically well investigated methods. Related applications exist as well for the recursive variant. Here we restrict our considerations on non-recursive RS-methods. Typical for the RSM is that simulation is used to generate a response curve. More exactly, a functional relation between the performance measure g (response surface) of a stochastic system and the decision variables is fitted to the response of a simulation model evaluated at several points for the decision variables. The resulting curve, also called *Meta model*, is assumed as a deterministic function and minimised in the optimisation step by a (deterministic) optimisation method. Such an approach needs extensive simulation experiments to get a well fitted Meta model. Often a *regression model* is used as Meta model, i.e., the Meta model is assumed as a polynomial function of the decision variables. As the rule are used polynomials of the first degree – linear models with linear response curve $g^{(RS)}(\theta) = A + B \cdot \theta$ - or polynomials of the second degree - parabolic functions $g^{(RS)}(\theta) = A + B \cdot \theta + C \cdot \theta^2$. Let us switch now to the formal description of RSM for the case when a linear regression model is used and $\theta \in R^1$. Starting point are n points (θ_i, y_i) in the plane, where $y_i = B(\theta_i, x_i)$, $i = 1..n$ (Fig.7). The θ_n's are called *controlled variables*. Their values are known exactly. The measured performance y_i, however, involves an *error* $e_i = y_i - (A + B \cdot \theta_i)$, which describes the distance of the measured performance y_i from the response curve. The idea now is to find such values A* and B* which minimise the *least square error* $LSE = (e_1^2 + \ldots + e_N^2)$.

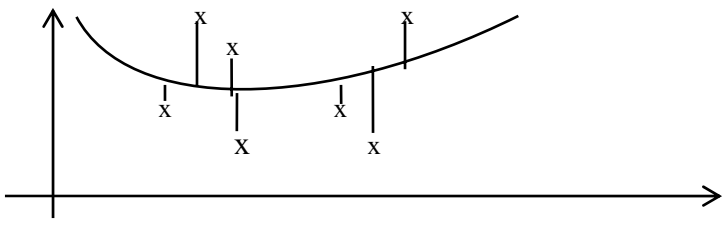


Figure 6. Example of a response curve

It is well-known (see PAPOULIS, 1990) that A* and B* are solutions of the following system of equations (we omit the indexation in the sums):

$$\left.\begin{aligned} n\ A + B \sum \theta_i &= \sum y_i \\ A \sum \theta_i + B \sum \theta_i^2 &= \sum \theta_i y_i \ . \end{aligned}\right\} \qquad (3.11)$$

In case of a parabolic function the optimal values A*, B*, and C* must be calculated from (see e.g. PAPOULIS, 1990, Ch.11).

$$\left.\begin{aligned} n\ A + B \sum \theta_i + C \sum \theta_i^2 &= \sum y_i \\ A \sum \theta_i + B \sum \theta_i^2 + C \sum \theta_i^3 &= \sum \theta_i y_i \\ A \sum \theta_i^2 + B \sum \theta_i^3 + C \sum \theta_i^4 &= \sum \theta_i^2 y_i \end{aligned}\right\} \qquad (3.12)$$

To demonstrate the RSM we use again the (S,S)-inventory problem. We start with the linear regression model. For that we assume that the inventory values $\theta_i = i$, i = 1..20, serve as measuring respectively experimental points and that in these points are simulated or observed the first 20 demand realisations from Table 2. For given inventory and given demand the corresponding cost we get from Equ.3.2. Thus the necessary for the RSM system outputs (responses) y_i can be calculated as $y_i = B(x_i, \theta_i)$, i = 1..20. For instance, it holds that $y_1 = B(3.12; 1) = p \cdot 2.12 = 6.36$. All 20 pairs (θ_i, y_i) are as follows:

(1; 6.36)	(2; 0.28)	(3; 25.47)	(4; 0.46)	(5; 4.22)	(6; 5.86)	(7; 6.40)
(8; 6.28)	(9; 8.24)	(10; 53.91)	(11; 10.8)	(12; 7.07)	(13; 45.39)	(14; 4.26)
(15; 5.66)	(16; 22.29)	(17; 16.83)	(18; 17.9)	(19; 15.30)	(20; 19.63) .	

With these data we calculate the sums in (3.11) and (3.12) as

$\sum \theta_i = 210$	$\sum \theta_i^2 = 2\,830$	$\sum \theta_i^3 = 44\,100$	$\sum \theta_i^4 = 722\,666$
$\sum y_i = 282.61$	$\sum \theta_i y_i = 3\,436.23$	$\sum \theta_i^2 y_i = 49\,183.75$	

and get for system (3.11) of equations the concrete form

$$\begin{aligned} 20\ A + 210\ B &= 282.61\ , \\ 210\ A + 2\,830\ B &= 3\,436.23\ . \end{aligned}$$

The solution is A* = 6.25424 und B* = 0.75012, such that we have the regression line

$$g^{(RS)}(\theta) = 6.25424 + 0.75012 \cdot \theta.$$

The minimum among all experimental points is reached for $\theta = 1$ with $g^{(RS)}(1) = 7.00436$. Obviously this result is far from the true solution and absolutely valueless (since the expected demand is 10 product units and the shortage cost are three times as much as the holding cost the optimal inventory level must be higher than the average demand). The only consequence is that the linear model is not suited for the inventory model. Let us switch to the parabolic model. With the same data we calculate from (3.12) the system of equations

$$\begin{aligned} 20\ A + 210\ B + 2\,830\ C &= 282.61\ , \\ 210\ A + 2\,830\ B + 44\,100\ C &= 3\,436.23\ , \\ 2\,830\ A + 44\,100\ B + 722\,666\ C &= 49\,183.75\ . \end{aligned}$$

The solution is A* = 24.29368526, B* = -3.394038528, and C* = 0.180055549. The minimising point θ* of the regression curve $g^{(RS)}(\theta) = A^* + B^* \theta + C^* \theta^2$ can be determined from $dg^{(RS)}/d\theta = 0$. We get $\theta^* = - B^* / (2 \cdot C^*)$. Because of the second derivative of $g^{(RS)}$ is equal to $d^2g^{(RS)}/(d\theta)^2 = C^*$, point θ* is minimising for $g^{(RS)}$ if and only if C* > 0, which is the case. Consequently, θ* = 9.424976172 is the optimal inventory level and for the minimal cost holds $g^{(RS)}(\theta^*) = 8.2993319077$. This result is better than the result for the linear model, but in comparison to the true solution still unsatisfactory.

To rate RSM with respect to its application in simulation optimisation we mention that the fitting technique and the design of the simulation experiment are critical features. There are however some serious disadvantages. To summarise we remark the following:

1. Well investigated statistical methods with acceptable user transparency exist for RSM.
2. RSM is a class of heuristic methods, where the criterion function g is approximated by a surface. To reach a sufficient approximation quality a considerable simulation effort is needed.
3. It must be suggested to apply RSM methods in a sequential way because of in any case only the local behaviour of the criterion function is well reflected.
4. RSM methods are local search methods and therefore applicable only for functions with a single optimum. However, even for simple problems as the optimisation of an (s, Q) inventory system the response surface shows a chaotic behaviour (cp. GREENWOOD et al., 1998).
5. Another problem is to define the search direction and the step size on the response surface.

For more information on RSM we refer to BARTON & MECKESHEIMER (2006). In the following chapters we consider the recursive SO approach.

4. The recursive principle of simulation optimisation

In Fig.3 of Section 2 we presented a scheme of the recursive SO, which is very general and contains only a few information how recursive SO works. Of cause there may be many realisations of that approach. In Fig.7 we show the principle of one suitable realisation. An ***optimiser*** gets an optimisation problem as input. A ***starter*** defines an initial decision. After that the search process will be realised through repeated processing of four stages – generation of relevant for the optimisation problem data by a ***simulator*** (realisation of a simulation experiment), an ***estimator*** uses the data from the simulation to estimate the value of the performance measure, a ***decider*** realises the task to accept the proposed decision or to continue the search process. In the latter case a ***searcher*** must propose other decisions. That cycle will be passed through until a stopping criterion will be fulfilled. The differences in relation to the scheme in Fig.5 are obvious. Once started the search process runs automatically without interaction of the user. After the search has stopped at least the best of all considered decisions will be returned, but it can be returned also the second best solution and so on. In general, as in the non-recursive case, two elements are needed – a simulator and an optimisation tool. As optimisation tool can serve arbitrary optimisation or search methods. We prefer Genetic Algorithms because of they have such advantages like independence of the application domain, suitability for very general optimisation problems, robustness with respect to initial decisions, they excellent deal with the random output of simulation experiments, they leave local optima and find the global one, and finally they need only a small amount of input information. For better understanding of the recursive approaches let us consider in the following chapter an applications of SO to a complex inventory problem.

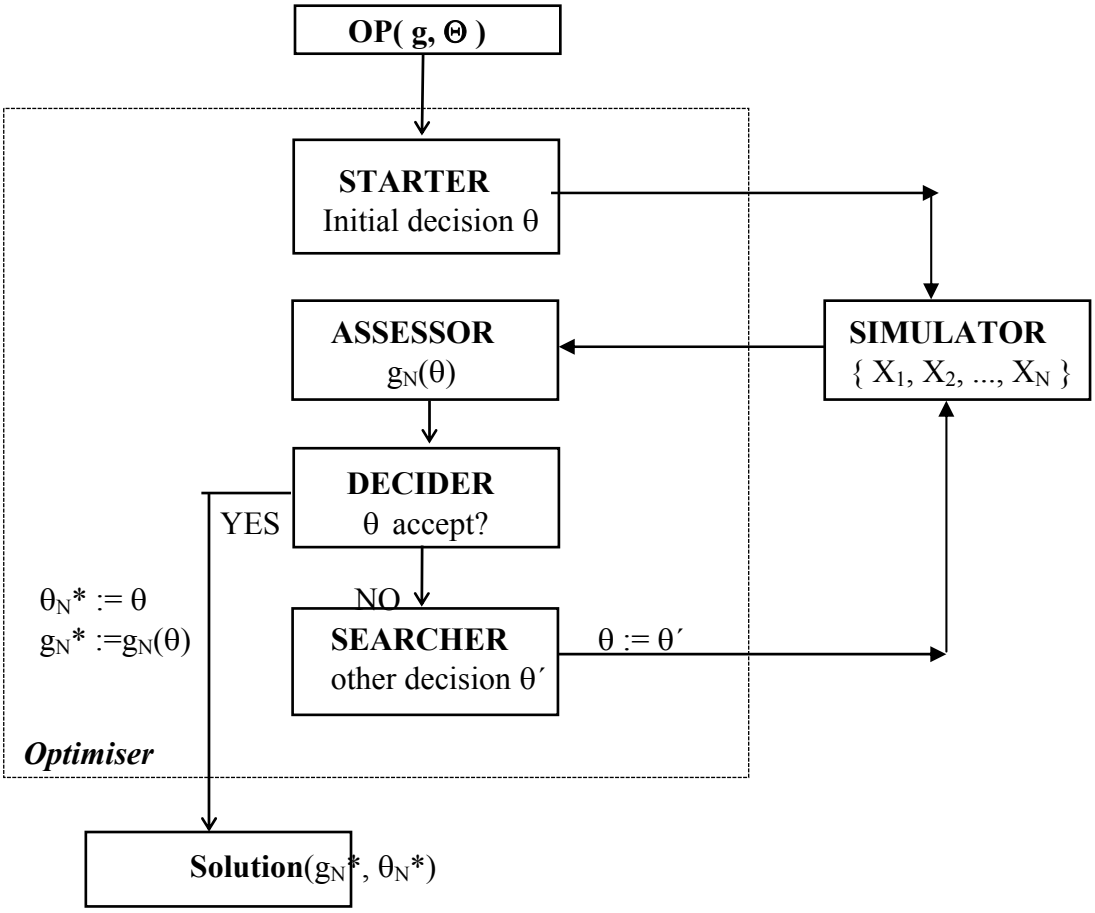


Figure 7. Scheme of the recursive simulation optimisation

5. Application of recursive simulation optimisation

In Example 2.5 of Chapter 2 was described a Multi-Location-Model with Transshipments (MLMT). The decision problem formulated there is an example for an optimisation problem that does not have an analytically tractable criterion function. Therefore the MLMT can serve as an excellent example for the recursive SO approach to demonstrate the basic idea and to understand the advantages and disadvantages of that approach. In the present chapter we first describe the model in more details, refer to some important results, and demonstrate our SO approach using Genetic Algorithms as optimiser. We also refer to some papers dealing with the same topic.

5.1. The MLMT-model

We assume that during a period $M \geq 2$ locations have to meet a random demand for the same single product. Within the period the following chronological order of events realises:

1. Ordering of product (the *ordering decision* OD) at the beginning of the period with instantaneous delivery and calculation of ordering cost.
2. Realisation of the random demand.
3. Satisfaction of demand in each location.
4. Reallocation of product still in stock (the *transshipment decision* TD) through instantaneous transshipments between locations.
5. Calculation of transshipment, holding, and penalty costs.

To model the just described scenario we will use the following notations:

R^M and R_+^M - the M-space respectively the non-negative orthant of M-space,

$\mathbf{x} = (x_1, x_2, ..., x_M) \in R^M$ - the vector of starting inventory positions before ordering, the *pre-ordering* inventory levels,

$\mathbf{a} = (a_1, a_2, ..., a_M) \geq \mathbf{x}$ - the vector of inventory positions after ordering, the OD or *post-ordering* inventory levels,

$\boldsymbol{\xi} = (\xi_1, \xi_2, \ldots, \xi_M)$ - the non-negative random demand vector,

$E(\boldsymbol{\xi}) = \boldsymbol{\mu} = (\mu_1, \ldots, \mu_M)$ - the expected demand with $0 < \mu_i < \infty$ for i=1..M,

$F(\cdot)$ and $f(\cdot)$ - the distribution respectively density function of demand vector $\boldsymbol{\xi}$,

$F_i(\cdot)$ and $f_i(\cdot)$ - the distribution resp. density function of the demand in location i,

$\mathbf{s} = (s_1, s_2, ..., s_M) \in R_+^M$ - the vector of demand realisation,

$\mathbf{y} = \mathbf{a} - \mathbf{s}$ - the vector of *pre-transshipment* inventory levels,

$\mathbf{b} = (b_{ij})_{i,j=1..M}$ - the transshipment decision TD, where b_{ij} denotes the amount of product to be transferred from location i to location j,

$\mathbf{z} = \mathbf{y} \oplus \mathbf{b}$ - the vector of *post-transshipment* inventory levels,[1]

k_i, h_i, p_i - ordering, holding, and penalty cost parameters per unit product in location i, i=1..M,

c_{ij} - transshipment cost per unit transferred from location i to location j; $i, j = 1..M$.

We assume $0 = c_{ii} \leq k_i, h_i, p_i, c_{ij} < \infty$, i, j=1..M, $i \neq j$.

The decision problem is to define such an OD and TD that minimise the expected cost of the system.

In general the just formulated decision problem does not have an analytical solution for M>2. To show this we concentrate in the remaining part of this subchapter on the single-period case. First of all, to describe the cost generated by each of the decisions, we introduce the two functions

$$k_1(\mathbf{x}, \mathbf{a}) = \sum_{i=1}^{M} k_i (a_i - x_i) \tag{5.1}$$

and

$$k_2(\mathbf{y}, \mathbf{b}) = k_2(\mathbf{a}, \mathbf{s}, \mathbf{b}) = \sum_{i=1}^{M} \left\{ h_i \cdot \max(z_i, 0) + p_i \cdot \max(-z_i, 0) + \sum_{j=1}^{M} c_{ij} b_{ij} \right\}. \tag{5.2}$$

Function $k_1(\cdot,\cdot)$ is connected with the OD and represents for given pair (**x**, **a**) the total order cost, whereas function $k_2(\cdot,\cdot)$, connected with the TD, describes for given pair (**y**, **b**) the total holding, shortage, and transshipment cost. It is meaningful for our considerations to express explicitly, that the vector **y** of pre-transshipment inventory levels depends on OD **a** and demand realisation **s**, i.e. $\mathbf{y} = \mathbf{a} - \mathbf{s}$. Thus we use $k_2(\mathbf{a}, \mathbf{s}, \mathbf{b})$ instead of $k_2(\mathbf{y}, \mathbf{b})$.

To describe the fact that we have to choose an OD and a TD for arbitrary pre-ordering respectively pre-transshipment inventory levels we introduce the notion *decision rule*. In particular, an *ordering rule* is a mapping $d_1: R^M \rightarrow R^M$ with $d_1(\mathbf{x}) \geq \mathbf{x}$ for $\mathbf{x} \in R^M$, i.e., an ordering rule d_1 assigns to each pre-ordering inventory level **x** a post-ordering inventory level $d_1(\mathbf{x})$. Condition $d_1(\mathbf{x}) \geq \mathbf{x}$ defines for given **x** the set of admissible ordering decisions, and means that the post-ordering inventory levels can not be smaller than the pre-ordering inventory levels. In other words, to sell product at the beginning of a period is not allowed.

[1] For the sake of conciseness, symbol $\oplus$ denotes for an M-vector u and MxM matrix $A = (A_{ij})$ the following operation: $u \oplus A = ((u \oplus A)_1, \ldots, (u \oplus A)_N)$ with $(u \oplus A)_i := u_i + \sum_{j=1}^{M} A_{ji} - \sum_{j=1}^{M} A_{ij}$ for all i.

Similarly a *transshipment rule* is a mapping $d_2: R^M \times R_+^M \rightarrow R^{M^2}$ with $d_2(\mathbf{a}, \mathbf{s}) \in \mathbf{B}(\mathbf{a}, \mathbf{s})$, $(\mathbf{a}, \mathbf{s}) \in R^M \times R_+^M$. Set $\mathbf{B}(\mathbf{a}, \mathbf{s})$ denotes for given $(\mathbf{a}, \mathbf{s})$ the set of admissible transshipment decisions, i.e.,

$$\mathbf{B}(\mathbf{a}, \mathbf{s}) = \{ \mathbf{b} = (b_{ij}): b_{ij} \geq 0, b_{ii} \geq \min(a_i, s_i), \sum_{j=1}^{M} b_{ij} = a_i, i, j = 1..M \}.$$

We remark that the assumed order of events implies $b_{ii} \geq \min(a_i, s_i)$, $i = 1..M$. A pair of decision rules $d = (d_1, d_2)$ is called a *policy*.
Now we realise backward induction and start with the TD, i. e., for each vector $(\mathbf{a}, \mathbf{s})$ we have to solve the *transshipment problem*

$$V_2(\mathbf{a}, \mathbf{s}) := \min_{\mathbf{b} \in \mathbf{B}(\mathbf{a}, \mathbf{s})} k_2(\mathbf{a}, \mathbf{s}, \mathbf{b}), (\mathbf{a}, \mathbf{s}) \in R^M \times R_+^M . \tag{5.3}$$

Function $V_2(\cdot,\cdot)$ represents the minimum total holding, shortage, and transhipment cost for given $(\mathbf{a}, \mathbf{s})$. We need also the expectation of function $V_2(\mathbf{a},\cdot)$ with respect to the demand, i.e.,

$$G_2(\mathbf{a}) = E_{\xi}[V_2(\mathbf{a}, \xi)] = \int_{\mathbf{s} \in R_+^N} V_2(\mathbf{a}, \mathbf{s}) \, dF(\mathbf{s}), \mathbf{a} \in R^M. \tag{5.4}$$

A *transhipment rule* $d_2^*: R^M \times R_+^M \rightarrow R^{M^2}$, which realises the minimum at the right-hand side of (5.3) respectively for which holds $k_2(\mathbf{a}, \mathbf{s}, d_2^*(\mathbf{a}, \mathbf{s})) = V_2(\mathbf{a}, \mathbf{s})$ for $(\mathbf{a}, \mathbf{s}) \in R^M \times R_+^M$, is called *minimising*.
By analogy we introduce function $V_1(\cdot)$ and define the *ordering problem* as

$$\begin{aligned} V_1(\mathbf{x}) &= \min_{\mathbf{a} \geq \mathbf{x}} \{ k_1(\mathbf{x}, \mathbf{a}) + E_{\xi}[V_2(\mathbf{a}, \xi)] \} = \min_{\mathbf{a} \geq \mathbf{x}} \{ k_1(\mathbf{x}, \mathbf{a}) + G_2(\mathbf{a})] \} = \\ &= \min_{\mathbf{a} \geq \mathbf{x}} \{ k_1(\mathbf{x}, \mathbf{a}) + \int_{\mathbf{s} \in R_+^M} \min_{\mathbf{b} \in \mathbf{B}(\mathbf{a}, \mathbf{s})} k_2(\mathbf{a}, \mathbf{s}, \mathbf{b}) \, dF(\mathbf{s}), \mathbf{x} \in R^M. \end{aligned} \tag{5.5}$$

$V_1(\cdot)$ denotes for given $\mathbf{x}$ the minimum of the expected total cost, i.e., the sum of ordering cost plus expected holding, shortage, and transhipment cost.
We define now a function $G_1(\cdot)$ by

$$G_1(\mathbf{a}) = \langle \mathbf{k}, \mathbf{a} \rangle + G_2(\mathbf{a}), \mathbf{a} \in R^M, \tag{5.6}$$

where vector $\mathbf{k} = (k_1, \ldots, k_M)$ and $\langle \cdot, \cdot \rangle$ denotes the usual scalar product.
From (5.5) and (5.6) follows that we can simplify the ordering problem to

$$V_1(\mathbf{x}) = \min_{\mathbf{a} \geq \mathbf{x}} \{ G_1(\mathbf{a}) \} - (\mathbf{k}, \mathbf{x}), \mathbf{x} \in R^M. \tag{5.7}$$

An *ordering rule* $d_1^*: R^M \rightarrow R^M$ is *minimising* if it realises the minimum in (5.7) for all $\mathbf{x} \in R^M$.
Finally, we formulate our decision problem in the following way:

For the single-period MLMT-model we have to find an optimal policy and to calculate the values $V_1(\mathbf{x})$ for all $\mathbf{x} \in R^M$.

From basic results of Markov decision theory (see e.g. HEYMAN & SOBEL, 1990) follows that the pair of minimising decision rules is an optimal policy $d^* = (d_1^*, d_2^*)$ for the here considered MLMT-model. However, we have no analytical tractable expression for function $V_2(\cdot)$ and $G_2(\cdot)$ and consequently also not for function $V_1(\cdot)$ and $G_1(\cdot)$. The reason for this is

connected with the transshipment problem. To compute in (5.3) for given pair (**a**, **s**) the minimising TD we have to solve a linear transportation problem with excess and shortage. But for such problems closed form solutions do not exist. One way to surmount the above-mentioned analytical problems is to look for structural properties of both the value and cost functions and the corresponding minimising decision rules. KÖCHEL (1975) has proven following results (see also ROBINSON, 1990):

Property 1. In case of linear ordering, holding, penalty, and transshipment cost functions it holds that function $G_1(\cdot)$, defined in (5.6), is convex in its arguments.

Property 2. In the MLMT-model with linear ordering, holding, penalty, and transhipment cost functions the minimising ordering rule is an order-up-to rule with order-up-to point $\mathbf{a}^* \in R_+^M$, where $\mathbf{a}^*$ is the global minimum point of function $G_1(\cdot)$.

Let us consider now structural properties of the minimising transhipment rule. The idea is to define such conditions on the cost parameters that the optimal transshipment rule avoids unreasonable transshipments. For given pair (**a**, **s**) we denote the set of surplus locations with positive pre-transshipment levels by $M^+(\mathbf{a}, \mathbf{s}) = \{i = 1..M: a_i > s_i\}$and the set of shortage locations with negative pre-transshipment inventory levels by $M^-(\mathbf{a}, \mathbf{s}) = \{i = 1 ..M: a_i < s_i\}$. Obviously, transshipments may go only from M^+ into all other locations. There are two cases.

Case 1: Transshipments from M^+ into M^- should be efficient, i.e., they should decrease total cost. This leads to the assumption "**E**fficiency of **T**ransshipments"

$$\text{(ET)} \qquad C_{ij} := h_i + p_j - c_{ij} > 0\ ,\ i=1..M.$$

Case 2: Transshipments from M^+ into $\{1, 2, ..., M\} \setminus M^-$ should be inefficient, i.e., they should increase total cost. We get the assumption "**R**elative **I**ndependence of the locations"

$$\text{(RI)} \qquad c_{ij} + h_j - h_i > 0\ , \qquad i, j=1..M, i \neq j.$$

Assumption (RI) means that transshipments between locations with positive inventory are unprofitable.

A third assumption is the "**S**hortest **W**ay" assumption

$$\text{(SW)} \qquad c_{ir} + c_{rj} > c_{ij}\ , \qquad i, j, r=1..M,\ i \neq j \neq r.$$

Assumption (SW) expresses that it is cheaper to transship directly than via another location.

Finally, to avoid unrealistic ordering policies we need assumption "**S**elf **O**rdering"

$$\text{(SO)} \qquad k_i + c_{ij} > k_j\ , \qquad i, j=1..M,\ i \neq j.$$

We can interpret assumption (SO) analogous to the previous assumption: It is cheaper to order directly than via another location.

Again KÖCHEL (1975) has shown

Property 3. Let the assumptions (ET), (RI), (SW), and (SO) be fulfilled. Then for the minimising transhipment rule holds

a) $b_{il} \cdot b_{lj} = 0$ ($i, j, l = 1..M$, $i \neq j \neq l$), i.e., no location can both initiate and receive transshipments.

b) $b_{ij} = 0$, $i=1..M$, $j \notin \mathbf{M}^-$, $i \neq j$, i.e., there are no transshipments to locations without shortage.

c) $\sum_{i \in \mathbf{M}^+} b_{ij} \leq s_j - a_j$, $j \in \mathbf{M}^-$,
i.e., the total transshipments to a shortage location do not exceed the shortage.

d) $$\sum_{i \in \mathbf{M}^+} \sum_{j \in \mathbf{M}^-} b_{ij} = \min \left[\sum_{i \in \mathbf{M}^+} (a_i - s_i); \sum_{j \in \mathbf{M}^-} (s_j - a_j) \right],$$
i.e., the total amount of transshipments is equal to the minimum of the total positive pre-transshipment inventory and the total negative pre-transshipment inventory.

The last property allows us to simplify $G_2(\cdot)$. Fix $\mathbf{a}$ and $\mathbf{s}$ and let $\mathbf{b}^* = (b_{ij}^*)$ denote the optimal TD. From (5.3) follows with property 3, that

$$V_2(\mathbf{a}, \mathbf{s}) = k_2(\mathbf{a}, \mathbf{s}, \mathbf{b}^*) = \sum_{i \in \mathbf{M}^+} h_i \left(a_i - s_i - \sum_{j \in \mathbf{M}^-} b_{ij}^*\right) + \sum_{j \in \mathbf{M}^-} p_j \left(s_j - a_j - \sum_{i \in \mathbf{M}^+} b_{ij}^*\right) + \sum_{i \in \mathbf{M}^+} \sum_{j \in \mathbf{M}^-} c_{ij} b_{ij}^* = \tag{5.8}$$

$$= \sum_{i \in \mathbf{M}^+} h_i (a_i - s_i) + \sum_{j \in \mathbf{M}^-} p_j (s_j - a_j) - \sum_{i \in \mathbf{M}^+} \sum_{j \in \mathbf{M}^-} C_{ij} b_{ij}^*,$$

and finally

$$G_2(\mathbf{a}) = \sum_{i=1}^{M} L_i(a_i) - C(\mathbf{a}), \mathbf{a} \in R^M, \tag{5.9}$$

with

$$L_i(a_i) = (h_i + p_i) \int_0^{(a_i)^+} F_i(s_i)\, ds_i + p_i (\mu_i - a_i), a_i \in R^1, i=1, \ldots, M, \tag{5.10}$$

and

$$C(\mathbf{a}) = \int_{\mathbf{s} \in R_+^M} \sum_{i \in \mathbf{M}^+} \sum_{j \in \mathbf{M}^-} C_{ij} b_{ij}^* f(\mathbf{s})\, d\mathbf{s}, \mathbf{a} \in R^M. \tag{5.11}$$

$L_i(a_i)$ represents the expected holding and penalty cost in location i with post-ordering inventory level a_i and without transshipments, and $C(\mathbf{a})$ represents for given $\mathbf{a} \in R^M$ the maximal expected cost savings from transshipments. Now formula (5.9) allows a simple interpretation: The value $G_2(\mathbf{a})$ is equal to the expected cost for independent locations minus the maximal expected gain (cost savings) from transshipments. Since $C_{ij} > 0$ by assumption (ET) we have $C(\mathbf{a}) \geq 0$ for all $\mathbf{a} \in R^M$. Consequently, for arbitrary given $\mathbf{x} \in R^M$ and $\mathbf{a} \in R^M$ the expected cost for a system with optimal transhipments, $G_1(\mathbf{a}) - (\mathbf{k}, \mathbf{x})$, do not exceed the cost for a system without transhipments, $L_1(a_1) + \ldots + L_M(a_M) + (\mathbf{k}, \mathbf{a} - \mathbf{x})$, i.e., transhipments save costs under assumption (ET).

Once more we want to point out the fact that despite the structural results on the optimal decision rules and the simplification of function $G_2(\cdot)$ in (5.9) generally there is still no analytical way to calculate $C(\mathbf{a})$ for given $\mathbf{a} \in R^M$. Consequently, traditional analytical approaches are not applicable.

Before we show in the following subchapter how to apply recursive simulation optimisation we briefly consider the non-recursive approach. Since for any OD **a** the value $G_2(\mathbf{a})$ from (5.4) is an expectation, $G_2(\mathbf{a}) = E_\xi[V_2(\mathbf{a}, \xi)]$, and since the sample average is an unbiased estimator for mean values the non-recursive approach gets the following concretisation:

(i) ***Simulation step***.

Through simulation of the demand (in accordance with the distribution function $F(\cdot)$ for ξ) generate a sample of realisations $\mathbf{s}^1$ to $\mathbf{s}^T$ with given size T and calculate for given **a** the estimation $G_{1,T}(\mathbf{a}) = (\mathbf{k}, \mathbf{a}) + [V_2(\mathbf{a}, \mathbf{s}^1) + \ldots + V_2(\mathbf{a}, \mathbf{s}^T)] / T$. We remark that the calculation of $V_2(\mathbf{a}, \mathbf{s})$ needs the solution of a linear transportation problem with excess and shortage.

(ii) ***Optimisation step***.

Apply an arbitrary method to find OD $\mathbf{a}^{(T)}$, which realizes the minimum for $G_{1,T}(\mathbf{a})$, and take $\mathbf{a}^{(T)}$ and $G_{1,T}(\mathbf{a}^{(T)})$ as approximations for the optimal OD $\mathbf{a}^*$ respectively for $G_1(\mathbf{a}^*)$. Since function $V_2(\cdot\ , \mathbf{s})$ is convex in **a** for each $\mathbf{s} \in R_+^M$, function $G_{1,T}(\cdot)$ is also convex for any sample. Consequently, the search for $\mathbf{a}^{(T)}$ is a convex minimisation problem.

To get an impression of the quality of the proposed procedure we consider a four-location model with exponentially distributed demand and $\boldsymbol{\mu}$ =(200, 300, 500, 400). The corresponding cost parameters are given in Table 4.

i	k_i	h_i	p_i	c_{i1}	c_{i2}	c_{i3}	c_{i4}
1	0	1	10	-	5	7	4
2	0	2	9	7	-	8	5
3	0	4	11	9	8	-	7
4	0	3	8	8	7	9	-

Table 4. Cost parameters for the 4-location MLIST-model

The exact solution is $\mathbf{a}^*$ = (663.00, 478.23, 566.80, 397.96) with $G_1(\mathbf{a}^*)$ = 4 467 monetary units (see KÖCHEL, 1977). We want remark that in the optimisation step was used coordinate-wise descent method to find the minimum of function $G_{1,T}(\mathbf{a})$, $\mathbf{a} \in R_+^M$. Table 5 contains the results for various sample sizes. As it was to be expected these results show the tendency that the approximation quality increases with increasing sample size T.

T	$a_1^{(T)}$	$a_2^{(T)}$	$a_3^{(T)}$	$a_4^{(T)}$	$G_{1,T}(\mathbf{a}^{(T)})$
100	625.05	477.83	611.97	371.95	4 890.93
200	721.53	487.67	558.14	333.62	4 446.92
1 000	627.47	473.92	580.88	393.39	4 348.73
2 000	675.99	474.00	561.50	387.84	4 423.18

Table 5. Results of the non-recursive approach for the 4-location MLMT-model

KÖCHEL (1998b) applied the non-recursive approach to the two-location MLMT. Whereas for the application of the non-recursive approach to the simple inventory problem from Example 2.3 we had to consider all T values of the demand sample (see §3.1) in the two-location MLMT all T^2 pairs of demand values in the two locations must be checked. Moreover, the optimal order up to point $\mathbf{a}_T^*$ is not necessary equal to one of these pairs. In principle we can apply the non-recursive approach to MLMT models with an arbitrary number N of locations, but we have to take into account that the numerical effort increases exponentially with the number N of locations.

ROBINSON (1990) suggests approximating the random demand in the MLMT by a finite discrete random variable, whereas TAGARAS & COHEN (1992) estimate the total cost for a two-location MLMT with replenishment lead times for a finite number of parameter values and take the best one.

5.2. Simulation optimisation of the MLMT-model – the recursive approach

In the present subchapter we return to the multi-period MLMT-model. We will explain how the combination of simulation and Genetic Algorithms (GAs) in a recursive manner can find good ordering and transhipment decisions. ARNOLD & KÖCHEL (1996) are the first who applied GAs to the MLMT-model. However, it is not obvious why we need a GA as an optimiser as long as the function to be minimised is convex. If in addition to the linear order cost parts we assume fixed ordering cost $K > 0$, which arise in case that at least one location orders a positive amount of product, then we have a completely new situation. Additionally we assume now an infinite planning horizon with the average long-run cost as performance measure. In contrast to the model in §5.1 there exist de facto no results on the optimal OD and TD, excluding a paper by HERER & RASHIT (1995), which have investigated the optimal OD in the single period, two location model. They have shown that the optimal OD has an extremely complicated structure, which is not suited for practical applications. If we add to this that usually in practice we will have more than two locations and therefore the mathematically true optimum leads of a certainty to ODs, which nobody can apply, then a good heuristic solution in case of fixed ordering cost will be the better decision. In the following we describe a way how to come to a good heuristic solution (cp. ARNOLD et al., 1997 and 1998).

First, we use that transhipment rule, which characteristics are described in Property 3 in §5.1. To get a meaningful ordering rule we proceed on the following three facts:

(1) From Property 2 in Subchapter 5.1 we know that for the MLMT without fixed order cost the optimal ordering rule is an order-up-to rule.
(2) In the single location model with fixed order cost part the optimal ordering rule has a (s, S)-structure (GRAVES et al., 1993).
(3) For the multi product model with fixed ordering cost part the optimal ordering rule has a ($\boldsymbol{\sigma}$, **S**)-structure (see e.g. KÜENLE, 1986).

In the light of these facts we recommend to concentrate on a ($\boldsymbol{\sigma}$, **S**)-ordering rule as the most promising and easy to implement rule. A ($\boldsymbol{\sigma}$, **S**)-ordering rule is defined as follows:

Let $\boldsymbol{\sigma} \subseteq R^M$ be a subset of R^M, $\mathbf{x} \in R^M$ be a vector of pre-ordering inventory levels, and $\mathbf{S} \in R^M$ an order-up-to vector. Then a ($\boldsymbol{\sigma}$, **S**)-ordering rule is a rule that chooses the vector $\mathbf{a} \in R^M$ of post-ordering inventory levels in accordance with the rule

```
IF  x ∈ σ
        THEN a := S          { OD orders up to S }
        ELSE  a := x .       { OD does not order }
```

How a ($\boldsymbol{\sigma}$, **S**)-ordering rule works is demonstrated in Figure 8, where in point $\mathbf{x}^{(1)}$ an ordering is realised but nothing is ordered in point $\mathbf{x}^{(2)}$.

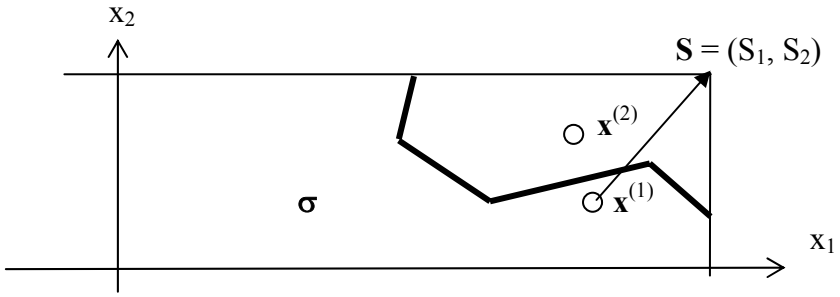


Figure 8. Action principle of a (σ, S)-ordering rule

The decision problem now is to define the optimal order-up-to vector $\mathbf{S}^* \in R_+^M$ and the optimal order region $\boldsymbol{\sigma}^* \subseteq R^M$. But we have analytical problems – first, we do not have an analytical expression for the performance measure, and second, to calculate set $\boldsymbol{\sigma}^*$ respectively its border line (more exactly the bold part in Fig. 8) is simply impossible. The first problem we overcome by simulation. To get a solution for the second problem we consider for a moment the single-period case with performance measure $G_1(\cdot)$ from (5.6). Then for the optimal order region holds $\boldsymbol{\sigma}^* = \{x \in R^M: K + G_1(\mathbf{S}^*) < G_1(x)\}$. The convexity property of function $G_1(\cdot)$ implies that the set $\underline{\boldsymbol{\sigma}}^* := \{\mathbf{x} \in R^M : \mathbf{x} \le \mathbf{S}^* \} \setminus \boldsymbol{\sigma}^*$ is a convex set. With that property in mind we suggest to approximate set $\boldsymbol{\sigma}^*$ by classes of special structured order regions such that set $\underline{\boldsymbol{\sigma}}^*$ is convex. For instance, the rectangular class of order regions is shown in Fig. 9. We can see that this class can be described by the vector $\mathbf{s} \in R^M$. For each $\mathbf{s} \in R^M$ the order region is given as

$$\boldsymbol{\sigma}(\mathbf{s}) = \{\mathbf{x} \notin \underline{\boldsymbol{\sigma}}(\mathbf{s})\} = \{\, \mathbf{x} = (x_1, ..., x_M): x_1 \le s_1 \vee x_2 \le s_2 \vee ... \vee x_M \le s_M\}.$$

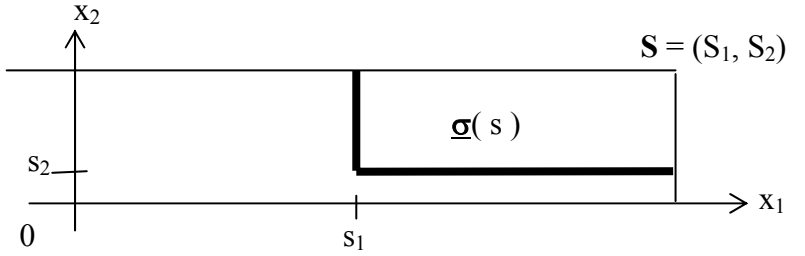


Figure 9. Principle of a rectangular ordering rule

The resulting ordering rule is obviously simple and easy to implement. But, the problem is how to calculate an optimal parameter vector $\mathbf{s}^*$? Before answering this important question we define some other classes of order regions. These are in detail the ellipse, circle, and triangular classes.

For the ellipse class the order region is defined by (cp. Fig.10)

$$\boldsymbol{\sigma}(\mathbf{s}) = \left\{ \mathbf{x} \le \mathbf{S} : \frac{(x_1 - S_1)^2}{(S_1 - s_1)^2} + ... + \frac{(x_M - S_M)^2}{(S_M - s_M)^2} > 1 \right\}.$$

As a special case of the ellipse class, when $(S_1 - s_1) = \ldots = (S_M - s_M) = r$, we get the circle class with order region

$$\boldsymbol{\sigma}(r) = \left\{ \mathbf{x} \le \mathbf{S} : \sum_{i=1}^{M} (x_i - S_i)^2 > r^2 \right\}.$$

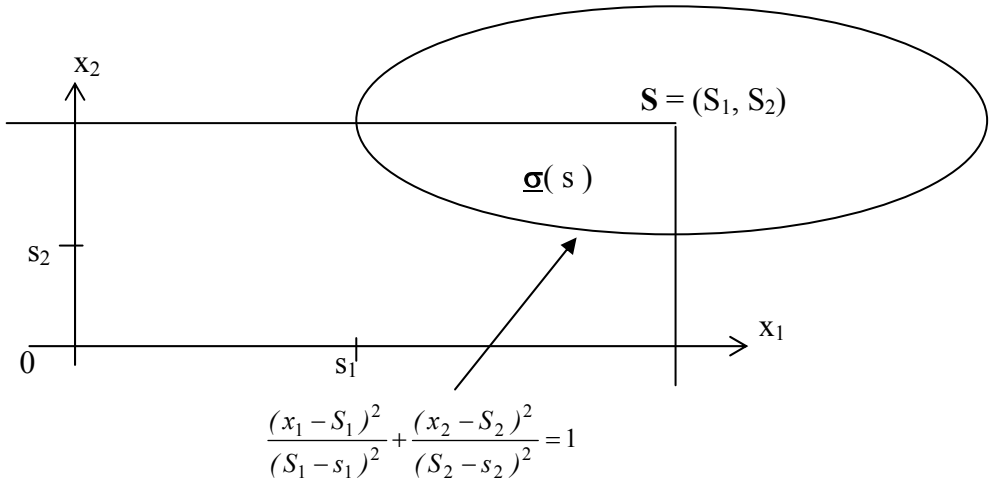


Figure 10. Principle of an ellipse ordering rule

In a similar way we can describe the order region for the triangular class (Fig.11). We get

$$\boldsymbol{\sigma}(\mathbf{s}) = \left\{ \mathbf{x} \leq \mathbf{S} : \sum_{i=1}^{M} \frac{x_i}{s_i} < 1 \right\}.$$

In all cases, excluding the circle case (which order region is described by the radius r), the order region is described by class type and a parameter vector **s**. Thus, introducing different classes of order regions, we have reduced the problem to find an optimal order region to the problem to choose one class and to find for that class the optimal parameter values. Finally we compare the best solutions of each class and take the best of them.

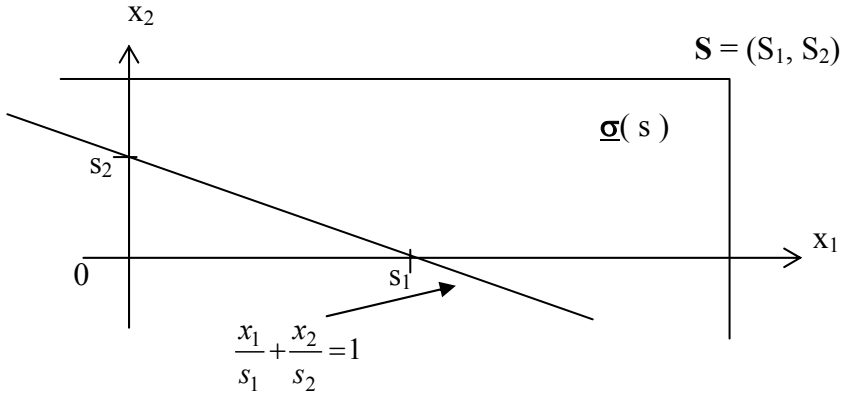


Figure 11. Principle of a triangular ordering rule

Basic requirement for the just formulated approach is that we can solve the corresponding parametric optimisation problems, where the performance measure $g(\boldsymbol{\sigma}(\mathbf{s}), \mathbf{S})$ denotes the long-run average cost under given $(\boldsymbol{\sigma}(\mathbf{s}), \mathbf{S})$ ordering rule. The place of parameter $\boldsymbol{\theta}$ in the general formulation of the optimisation problem in Chapter 2 is now taken by a $(\boldsymbol{\sigma}(\mathbf{s}), \mathbf{S})$ ordering rule. To find optimal $(\boldsymbol{\sigma}(\mathbf{s}), \mathbf{S})$-values we combine simulation and Genetic Algorithms. We need simulation to estimate the values of the performance measure, whereas the use of Genetic Algorithms as optimiser is based on a number of advantages described already in Chapter 2. The information interchange between the simulator and the Genetic Algorithm is very simple. The simulator gets a parameter vector from the Genetic Algorithm, i.e., the class of the ordering rule and values for the components of the two vectors **s** and **S**.

The simulator estimates by a simulation experiment the performance, returns these values to the Genetic Algorithm, which on the base of performance estimations decides to stop the search process or to send new parameter vectors to the simulator. Let us first look at the simulation, which can be described by the following algorithm.

Algorithm MLMT-SIMULATION {to get an estimation $G_{av}(\mathbf{s}, \mathbf{S})$ for $g(\sigma(\mathbf{s}), \mathbf{S})$}

INPUT: Number *M* of locations; number *T* of periods to simulate; cost parameters; demand parameters; initial inventory $\mathbf{x}(1)$; ordering rule class and parameters $\mathbf{s}$ and $\mathbf{S}$.

1. INITIALISATION: $t := 1$; $x_i := x_i(1)$, $i=1..M$.

2. ORDER DECISION : Calculation of ordering cost

$$G_{order}(t) := \begin{cases} K + \sum_{i=1}^{M} k_i \cdot (S_i - x_i)\,, x \in \sigma(s); \\ 0 \qquad\qquad\quad , else \end{cases}$$

and inventory levels

$$a_i := \begin{cases} S_i\,, x \in \sigma(s); \\ x_i\,, else. \end{cases}$$

3. DEMAND realisation: $\xi \rightarrow \mathbf{d} = (d_1, d_2, \ldots, d_M)$.

4. TRANSSHIPMENT DECISION: Optimal reallocation $\boldsymbol{b}^* = (b_{ij}^*)$ and calculation of transshipment cost

$$G_{trans}(t) := \sum_{j=1}^{M}\sum_{j=1}^{M} c_{ij} \cdot b_{ij}^*,$$

calculation of post-transshipment inventory levels

$$x_i := \begin{cases} a_i - d_i - \sum_{j=1}^{M} b_{ij}^*\,, d_i \le a_i; \\ a_i - d_i + \sum_{j=1}^{M} b_{ji}^*\,, d_i > a_i, \end{cases}$$

and inventory cost

$$G_{inv}(t) := \sum_{i=1}^{M}\left[h_i \cdot \max(0; a_i - d_i - \sum_{j=1}^{M} b_{ij}^*) + p_i \cdot \max(0; d_i - a_i - \sum_{j=1}^{M} b_{ji}^*) \right].$$

5. STOPPING CRITERION: IF ($t = T$) THEN (go to OUTPUT)
ELSE ($t := t+1$ and go to step 2).

OUTPUT: Estimation $G_{av}(\mathbf{s}, \mathbf{S}) := 1/T \cdot \sum_{t=1}^{T}\left[G_{order}(t) + G_{trans}(t) + G_{inv}(t)\right]$.

Next we give a brief introduction to Genetic Algorithms (GA). Typical for a GA is that he works with a set of solutions, the so-called population, which is moved through the parameter space. That movement is realised applying several genetic operators like selection, crossover,

and mutation to the individual solutions. The operators itself also depend on different parameters. We will explain it below. But first we introduce some definitions, which are used in connection with a GA and which originate from genetics.

Θ	-	the parameter space
solution $\boldsymbol{\theta} \in \Theta$	-	an *individual*
component θ_i	-	a *gene*
given value of a gene	-	an *allele*
subset $U \subset \Theta$ of solutions	-	a *population*
P	-	the *population size*
Γ	-	the *number of generations*
mapping $G: \Theta \rightarrow R^1$	-	the *fitness function*
$G(\boldsymbol{\theta})$	-	the *fitness* of individual $\boldsymbol{\theta}$

GAs solve optimisation problems of a great variety. A general formulation in terms of GAs is *that we must find an individual* $\boldsymbol{\theta}^* \in \Theta$ *with maximal fitness*, i.e., an individual for which holds $G(\boldsymbol{\theta}^*) \geq G(\boldsymbol{\theta})$ for all $\boldsymbol{\theta} \in \Theta$. All existing GAs realise in principle the following algorithm.

Algorithm EO {basic steps in ***E***volutionary ***O***ptimisation}

INPUT: Population size $P < \infty$; stopping criteria; fitness function G.

1. INITIALISATION: Chose P individuals from Θ for a starting population.

2. COMPUTE-step: Compute or estimate the fitness of all individuals of the actual population.

3. PRODUCTION-step: {application of genetic operators to produce the next generation}

- *3.1. Selection-step*: Chose individuals from the actual population.
- *3.2. Genetic-operation-step*: Apply genetic operators to the selected individuals and produce new individuals.
- *3.3. Production termination*: IF (number of new produced individuals < P) THEN (go to step 3.1).

4. STOPPING CRITERION: IF (stopping criterion is fulfilled) THEN (go to OUTPUT) ELSE (go to step 2).

OUTPUT: Individual with the best fitness and its fitness-value.

It is obvious that the general formulation of algorithm EO allows a great variety of concrete implementations. Let us begin with the stopping criterion. The simplest criterion is to stop when a maximal number Γ_{max} of generations is produced. Another way is to fix a small number $\varepsilon > 0$ and a number Γ_{stop} of generations and to stop the search process if during Γ_{stop} generations no improvement of at least ε for the best fitness could be observed. To explain some ideas how to concretise the genetic operators we return to the MLMT-model.
In case of the rectangular, triangular, and ellipse classes it holds that $\boldsymbol{\theta} = (\mathbf{s}, \mathbf{S}) \in R^{2M}$. For the fitness function holds $G(\boldsymbol{\theta}) = - G_{av}(\mathbf{s}, \mathbf{S})$, where $G_{av}(\mathbf{s}, \mathbf{S})$ is delivered by algorithm MLMT-SIMULATION. Let $\boldsymbol{\theta}_1$ to $\boldsymbol{\theta}_P$ denote the individuals of the actual population. A common selection principle is fitness-proportional selection, where the probability for an individual to

be selected equals the proportion of its own fitness and the total fitness of all individuals of a population. This principle is directly applicable in case that all fitness values are non-negative. However, in the MLMT-model all fitness values are non-positive. In those cases we must transform the fitness values $G(\boldsymbol{\theta})$ to non-negative values $G^{+}(\boldsymbol{\theta})$. For this let $\boldsymbol{\theta}_{worst}$ define the individual of the actual population with the highest cost $G(\boldsymbol{\theta}_{worst})$. Then for all individuals $\boldsymbol{\theta}$ of the actual population we set $G^{+}(\boldsymbol{\theta}) = G(\boldsymbol{\theta}) - G(\boldsymbol{\theta}_{worst})$.
After selection the operators crossover and mutation are applied to correspondingly selected individuals. Let $\boldsymbol{\theta}^1 = (\mathbf{s}^1, \mathbf{S}^1)$ and $\boldsymbol{\theta}^2 = (\mathbf{s}^2, \mathbf{S}^2)$ two selected individuals. The simplest crossover is the one-point crossover. Applying it to the two individuals $\boldsymbol{\theta}^1 = (s_1^1, \ldots, s_M^1; S_1^1, \ldots, S_M^1)$ and $\boldsymbol{\theta}^2 = (s_1^2, \ldots, s_M^2; S_1^2, \ldots, S_M^2)$ means that at random (e.g. with equal probability) a number between 1 and $2M$ is chosen. Let this number be $n < M$. Then two new (child) individuals are produced by changing the corresponding parts of the (parent) individuals (see Fig.12).

$$\boldsymbol{\theta}^1 = (s_1^1, \ldots, s_n^1 \,|\, s_{n+1}^1, \ldots, s_M^1; S_1^1, \ldots, S_M^1) \rightarrow \boldsymbol{\theta}'^1 = (s_1^2, \ldots, s_n^2, s_{n+1}^1, \ldots, s_M^1; S_1^1, \ldots, S_M^1)$$

$$\boldsymbol{\theta}^2 = (s_1^2, \ldots, s_n^2 \,|\, s_{n+1}^2, \ldots, s_N^2; S_1^2, \ldots, S_M^2) \rightarrow \boldsymbol{\theta}'^2 = (s_1^1, \ldots, s_n^1, s_{n+1}^2, \ldots, s_N^2; S_1^2, \ldots, S_M^2)$$

Figure 12. The scheme of one-point crossover

Whereas crossover is a sexual operator mutation is an asexual operator, i.e., mutation is applied to a single individual. Again there exist manifold concretisations of the general principle, which says to chose at random a gene (for instance S_5, if $M \geq 5$) of individual $\boldsymbol{\theta}$ and to change the allele a "little bit". Thus we can add to the value of S_5 the value of a realisation of a normally distributed random variable with zero mean and small variance.

The just described approach, which first application to MLMTs is presented in ARNOLD et al. (1997), is suited to solve arbitrary parametric optimisation problems. For instance KÄMPF & KÖCHEL (2006) considered a multi-product production-and-inventory model and applied a GA to define as well optimal parameters of (s, Q) ordering policies for each product as the optimal sequencing of products in the production stage. Another application of GAs to an (s, S) inventory model with random and constant lead times is realised by DENGIZ et al. (1997) respectively MAK et al. (1999). Besides GAs one can find applications to (s, S) models of Pattern Search and Nelder-Mead Search (HADDOCK & BENGU, 1987). LOPEZ-GARCIA & POSADA-BOLIVAR (1999) applied Tabu Search to (S, Q), (s, S), (nQ, s, T), (S, T), and (s, S, T) inventory models with the average total cost criterion. BASHYAM & FU (1994) apply Perturbation Analysis to the (s, S) model. With respect to the MLMT-model we want to point to HERER et al. (2004), which applied Infinitesimal Perturbation Analysis as a sample path optimisation technique.
Multi-echelon models with linear, assembly, and tree structures investigate GLASSERMAN & TAYUR (1995). They applied simulation and Perturbation Analysis to estimate sensitivities of inventory cost with respect to policy parameters. KÖCHEL & NIELÄNDER (2005) combine simulation and GAs for the optimisation of Multi-echelon models with arbitrary structure. During the last time corresponding research at the chair Modelling and Simulation moved into two directions. Firstly, we generalised the MLMT presented here with regard to several aspects. Major generalisations are related to the cost functions and the control process. In the MLMT of §5 we assumed linear cost functions and discrete time control (periodic review) both for ordering and transhipment decisions. However, usually in reality we have discrete review for ordering and continuous review for transhipments as well as non-linear cost functions, at least some fixed ordering costs. In KÖCHEL & HOCHMUTH (2009) is described a corresponding simulator. Furthermore, that simulator can handle in fact

arbitrary demand processes. It is allowed that the demand process for any location i can be **compound renewal**. Such a process is described by two independent random variables (r.v.) T_i and B_i for the inter-arrival time of clients at location i respectively their demand, $i = 1..M$. Compound renewal demand processes model real situations in a more adequate manner. They model both the arrival moments and the demand values of all clients. Thus exact holding and penalty costs can be calculated. Up to now in models with discrete review the whole demand of a period is transformed to the end of the period (cp. §5). In the consequence we have else holding or penalty costs. But in reality at the beginning of a period there will be some on-hand inventory with corresponding cost and only after some time is elapsed shortage is possible with corresponding penalty cost. That disadvantage of periodic review models is not existent in continuous review models. Secondly, we used other heuristic search methods than GAs. For instance, KÖCHEL & THIEM (2008) investigated how swarm intelligence can be used to solve complex inventory and logistic problems.
These are only some papers, which are dealing with SO of stochastic systems. At present exists a magnitude of application areas for SO. However, as a rule the considered systems are simple and far from reality. To handle realistic models corresponding simulators are necessary. At present time this is the major obstacle for a broader application of SO. Some ideas how to overcome these problems are discussed in the next section.

6. A framework for simulation optimisation applications

Notwithstanding the developments in computer techniques to realise SO is still a time- and labour-consuming task. Therefore some suggestions, which are based partially on our own experiences, on how to handle the application of SO may be helpful. It makes sense to distinguish between requirements on the necessary software and some kind of general advices. We start with the latter. Some hints for a methodical practice we can deduce from the previous chapter.

1. Realise *analytical investigations of the problem*. If it is not possible for the primary problem then consider else an approximate or simpler problem. The goal is to get information e.g. on
   - favourable decisions,
   - structural properties of the solution (convexity, monotonicity, etc.),
   - the granularity of the problem.
2. Restrict the search for an optimal solution to solution classes with a simple structure, well suited for practical applications, and try to parameterise the problem.
3. Define the search-set (by bounds etc.) of meaningful values for the input parameters.
4. Develop a SO pre-processor or starter to suggest e.g. starting points for optimisation.
5. Apply hybrid algorithms, which combine several optimisation methods like Genetic Algorithms, Tabu Search, Simulated Annealing and others into a single optimiser. In all our applications hybrid optimisers have outperformed other ones.

Unfortunately the actual situation is so that the biggest part of time expenditure for an SO project is spend rather for software design purposes than for model development and problem analysis. To improve this situation suited software tools with corresponding properties must be developed and implemented. Necessary are

- tools to support the ***modelling*** of (inventory) systems with such main elements like stores, products, workers, handling and transportation equipments, demand processes, control policies (ordering, release, transshipments, …), etcetera;
- tools to support (in an adaptive way) ***simulation experiment design*** for the choice of parameter vectors, number of replications, run length, transition phase, and problem-oriented processing of experimental data;

- ***optimisation tools*** for several search procedures (finite, discrete, continuous search space) with the possibility to combine various search procedures, with an user support for the choice of search procedures and also with several "non-search" generators e.g. for manually user input.

Important are ***interfaces***, which realise the separation of simulation and optimisation, such that it is easy to incorporate new simulators and optimisation procedures, but also user actions and problem-specific procedures respectively restriction to some procedures. Finally a ***knowledge and learning component*** will help to use problem-specific procedures and to decrease the total time required for a simulation optimisation project. Some more ideas on which properties SO software should have one can found for example in BOESEL et al. (2003).

At Chemnitz University of Technology we have done a few steps in that direction. For optimisation purposes the two tools CHEOPS (*C*hemnitzer *H*ybrid-*E*volutionäres *OP*timierungs*S*ystem, NIELÄNDER, 2009) and CAOS (*C*alculation *A*ssessment *O*ptimisation *S*ystem, see KÄMPF, 2008) are developed. CHEOPS makes available a multitude of elements for Genetic Algorithms, such that it supports the construction of problem-adequate Genetic Algorithms. In the system CAOS are implemented a Genetic Algorithm, Tabu Search and Simulated Annealing, which all may be used in an isolated manner or altogether as a hybrid optimiser in a sequential or parallel version. For these software tools are defined simple interfaces that allow combining them with arbitrary simulators. We want to point here to the simulator KaSimIR for the simulation of Kanban systems (see KÖCHEL et al., 2001; KÖCHEL & NIELÄNDER, 2002), which is also suited to simulate multi-echelon inventory systems (see KÖCHEL & NIELÄNDER, 2005), and to a new MLMT-simulator (KÖCHEL & HOCHMUTH, 2009).

7. Conclusion

We have shown that for a great variety of (inventory) problems, if not for any, sufficiently good solutions can be obtained by SO. In more detail we considered two inventory problems related to the (S, S) and the MLMT models and demonstrated the application of the non-recursive and the recursive SO approaches. For the recursive approach we briefly discussed the use of Genetic Algorithms as a general applicable optimisation tool. Whereas the theoretical base for different approaches in SO is well developed real applications in practice are very rarely. The main reason for that are the deficiencies in existing software and the necessary big amount of resources to realise SO. Therefore future research in our opinion should be directed to three main fields:

1. Substantial reduction of computing time to realise a simulation optimisation study. This can be done as well through parallel and distributed computing as through a good planning of experiments. First experiences are reported in KÖCHEL & RIEDEL (2004).
2. Further improvement of SO software towards a better support of users with different knowledge at least into two directions – to build simulation models (modelling phase) and to plan simulation experiments (execution phase). A big step would be the definition of a set of building elements for corresponding systems respectively models and the implementation of a corresponding kit.
3. Broadening of the application area from as a rule classical problem formulations to actual ones as for instance reverse logistics and supply chain management.

Further success will depend on how will be organised the collaboration between operations researchers, (business) economists, and computer scientists.

References

ARNOLD, J. & KÖCHEL, P. (1996). Evolutionary optimisation of a multi-location inventory model with lateral transhipments. *9th International Working Seminar on Production Economics*, Igls 1996, Pre-prints 2, 401-412.

ARNOLD, J. , KÖCHEL, P., UHLIG, H. (1997). Simulation und Evolutionäre Optimierung eines Mehrlagersystems mit Transport. In Biethahn, J. (Ed.). *Proceedings zum 6. Symposium: Simulation als betriebliche Entscheidungshilfe,* Göttingen 1997, 135-148

ARNOLD, J., KÖCHEL, P., UHLIG, H. (1998). With Parallel Evolution towards the Optimal Order Policy of a Multi-location Inventory with Lateral Transshipments. In: Papachristos, S. & Ganas, I. (eds.), Inventory Modelling in Production and Supply Chains. Research Papers of the Third ISIR Summer School, Ioannina, Greece, 1998, 1-14

BARTON, R.B. & MECKESHEIMER, M. (2006). Metamodel-Based Simulation Optimization. Chapter 18 in *Handbooks in Operations Research and Management Science, Vol. 13: Simulation*, S.G. Henderson and B.L.Nelson, eds., Elsevier

BASHYAM, S. & FU, M.C. (1994). Application of perturbation analysis to a class of periodic review (s, S) inventory systems. *Naval Research Logistics*, **41**, 47-80.

BOESEL, J., NELSON, B.L., ISHII, N. (2003). A framework for simulation-optimization software. *IIE Transactions*, **35**, 221-229.

DENGIZ, B., SEN, F., BULGAK, A. (1997). Optimization of stochastic systems using genetic algorithms. *Transactions on Operational Research*, **9**, 43-62.

FU, M.C. (1994.a). Optimization via simulation: A review. *Annals of Operations Research*, **53**, 199-247

FU, M.C. (1994.b).Sample Path Derivatives for (s, S) Inventory systems. *Annals of Operations Research* 53 (1994), 351-364

FU, M.C. (2002). Optimization for simulation: Theory vs. practice. *INFORMS Journal of Computing*, **14**, 192-215.

FU, M.C., GLOVER, F.W., and APRIL, J. (2005). Simulation optimization: A review, new developments, and applications. In *Proceedings of the 2005 Winter Simulation Conference*, 83-95.

FU, M.C. & HEALY, K. (1997). Techniques for optimisation via simulation: An experimental study on an (s, S) inventory system. *IIE Transactions*, **29**, 191-199.

GRAVES,S., RINNOOY Kan, A., and ZIPKIN, P. (Eds.) (1993). *Handbooks in Operations Research and Management Science, Vol. 4, Logistics of Production and Inventory.* North Holland

GREENWOOD, A.G., REES, L.P., and SIOCHI, F.C. (1998). An investigation of the behavior of simulation response surfaces. *European Journal of Operational Research*, **110**, 282-313.

GLASSERMAN, P. & TAYUR, S. (1995). Sensitivity Analysis for Base-stock Levels in Multi-echelon Production-inventory Systems. *Management Science*, **41**, 263-281.

HADDOCK, J. & BENGU, G. (1987). Application of a simulation optimisation system for a continuous review inventory model. In *Proceedings of the 1987 Winter Simulation Conference,* 382-390.

HADER, S. (1998). Das wissensbasierte Optimierungssystem DIM_EXPERTE. Forschungsbericht des Innovationskollegs „Bildung eines vernetzten Logistik- und Simulationszentrums“, TU Chemnitz 1998

HADER, S. (2001). Ein hybrider Ansatz zur Optimierung technischer Systeme. Dissertation TU Chemnitz. Publicised by Shaker Verlag, Aachen

HEALY, K.J.; SCHRUBEN, L.W. (1991). Retrospective simulation response optimization. In *Proceedings of the 1991 Winter Simulation Conference*, Nelson, B.L., Kelton, W.D. and Clark, G.M. (eds.),IEEE, Piscataway, NJ, pp. 901-906

HERER, Y.; ÖZDEMIR, D.; YUCESAN, E. (2004). A Monte-Carlo Simulation Approach to the Capacitated Multi-location Transshipment Problem. *Proceed. of the 2003 Winter Simulation Conference*, eds. S.Chick, P.J.Sanchez, D.Ferrin, and D.J.Morrice. 1729-1736. Piscataway, New Jersey: Institute of Electrical and Electronics Engineering

HERER, Y. & RASHIT, A. (1995). Lateral Stock Transshipments in a Two-location Inventory System. Proceed. of The 13th Int. Conf. on Prod. Res., Jerusalem 1995, 593-595.

HEYMAN, D.P., SOBEL, M.J. (Eds.) (1990). *Handbooks in Operations Research and Management Science, Vol. 2, Stochastic Models.* North Holland

HOCHMUTH, Ch. A. & KÖCHEL, P. (2008). A Simulator for Logistic Systems with Hub-and-spoke Structure. In Ivanov, D., Jahns, C., Struabe, F., Procenko, O., Sergeev, V. (eds.):Logistics and Supply Chain Management: Trends in Germany and Russia. Publishing House of the Saint Petersburg State Polytechnic University, 431-442

KÄMPF, M. (2008). *Software-Framework zur Simulationsbasierten Optimierung mit Anwendung auf Produktions- und Lagerhaltungssysteme*. Dissertation, Chemnitz University of Technology, Chemnitz 2008

KÄMPF, M. & KÖCHEL, P. (2006). Simulation-based sequencing and lot size optimisation for a production-and-inventory system with multiple items. *International Journal of Production Economics*, **104**, 191-200.

KÖCHEL, P. (1975). A Stochastic Inventory Model for Some Interconnected Locations (In German). Mathematische Operationsforschung und Statistik, **6**, 413-426. For the English version see http://www.tu-chemnitz.de/informatik/ModSim/publikationen.php

KÖCHEL, P. (1977). About the Optimal Inventory Control in a System of Locations: An Approximate Solution (In German). Mathematische Operationsforschung und Statistik, Serie Optimisation, **8**, 105-118.

KÖCHEL, P. (1982). Ein dynamisches Mehr-Lager-Modell mit Transportbeziehungen zwischen den Lagern. *Mathem. Operationsforsch. Statistik, Ser. Optimization,* 13, 267-286

KÖCHEL, P. (1998a).A Survey on Multi-location Inventory Models with Lateral Transshipments. In: Papachristos, S. & Ganas, I. (eds.), Inventory Modelling in Production and Supply Chains. Research Papers of the Third ISIR Summer School, Ioannina, Greece, 1998, 183-207

KÖCHEL, P. (1998b). Retrospective optimisation of a two-location inventory model with lateral transshipments. In *Proceedings of the 2nd International Conference on Traffic Science – ICTS '98*, 129-139

KÖCHEL, P. & HOCHMUTH, Ch. (2009). Optimisation experiments with a new simulator for inventory systems with multiple locations and lateral transshipments. *In print*.

KÖCHEL, P.; KUNZE, S.; NIELÄNDER, U., 2003. Optimal control of a distributed service system with moving resources: Application to the fleet sizing and allocation problem. Internat. Journal of Production Economics, v.81-82, 443-459

KÖCHEL, P. & NIELÄNDER, U. (2002). Kanban optimization by simulation and evolution. *Production Planning & Control*, **13**, 725-734

KÖCHEL,P. & NIELÄNDER, U.(2005). Simulation-based optimisation of Multi-echelon inventory systems. *International Journal of Production Economics*, **93-94**, 505-513.

KÖCHEL, P. , NIELÄNDER, U., Sturm, M. (2001). KASIMIR – object-oriented KAnban SIMulation Imaging Reality. Chemnitzer Informatik-Berichte, CSR-01-03

KÖCHEL, P. & RIEDEL, M.: *Performance Evaluation of Parallel Genetic Algorithms for Optimization Problems of Different Complexity.* Parallel Computing: Software Technology, Algorithms, Architectures and Applications. Advances in Parallel Computing, v. 13, Elsevier 2004, 313-320

KÖCHEL, P. & THIEM, ST. (2008). Search for good policies in a single-warehouse, multi-retailer system by particle swarm optimisation. Submitted to *International Journal of Production Economics*

KÜENLE, H.-U. (1986). Stochastische Spiele und Entscheidungsmodelle. Teubner Texte zur Mathematik, vol. 89, Leipzig

LOPEZ-GARCIA, L. & POSADA-BOLIVAR, A. (1999). A Simulator that uses Tabu Search to Approach the Optimal Solution to Stochastic Inventory Models. *Computers & Industrial Engineering*, **37**, 215-218

MAK, K.L., WONG, Y.S., HUANG, G.Q. (1999). Optimal Inventory Control of Lumpy Demand Items Using Genetic Algorithms. *Computers & Industrial Engineering*, **37**, 273-276

NIELÄNDER, U.(1999). Simulation Optimisation of Kanban Systems using a non-standard Genetic Algorithm. 4th ISIR Research Summer School on Inventory Modeling, Exeter, UK.

NIELÄNDER, U.(2009). CHEOPS – Das **C**hemnitzer **H**ybrid-**E**volutionäre **OP**timierungs**S**ystem. Dissertation, Chemnitz University of Technology 2009

PAPOULIS, A. (1990). Probability & Statistics. Prentice-Hall International, Inc.

PFLUG, G.Ch. (1996). Optimization of stochastic models. The Interface Between Simulation and Optimization. *Kluwer Academic Publishers*, Boston/Dordrecht/London

ROBINSON, L.W. (1990). Optimal and Approximate Policies in Multi-period Multi-location Inventory Models with Transshipments. *Oper. Res.* 38, 278-295

TAGARAS, G. & COHEN, M.A. (1992). Pooling in Two-location Inventory Systems with non-negligible Replenishment Lead Times. *Management Science* 38, 1067-1083.

TEKIN, E. & SABUNCOUGLU, I. (2004). Simulation optimisation: A comprehensive review on theory and applications. *IIE Transactions*, **36**, 1067-1081.

Chemnitzer Informatik-Berichte

In der Reihe der Chemnitzer Informatik-Berichte sind folgende Berichte erschienen:

CSR-04-02 Petr Kroha, Ricardo Baeza-Yates, Classification of Stock Exchange News, November 2004, Chemnitz

CSR-04-03 Torsten Hoefler, Torsten Mehlan, Frank Mietke, Wolfgang Rehm, A Survey of Barrier Algorithms for Coarse Grained Supercomputers, Dezember 2004, Chemnitz

CSR-04-04 Torsten Hoefler, Wolfgang Rehm, A Meta Analysis of Gigabit Ethernet over Copper Solutions for Cluster-Networking, Dezember 2004, Chemnitz

CSR-04-05 Christian Siebert, Wolfgang Rehm, One-sided Mutual Exclusion A new Approach to Mutual Exclusion Primitives, Dezember 2004, Chemnitz

CSR-05-01 Daniel Beer, Steffen Höhne, Gudula Rünger, Michael Voigt, Software- und Kriterienkatalog zu RAfEG - Referenzarchitektur für E-Government, Januar 2005, Chemnitz

CSR-05-02 David Brunner, Guido Brunnett, An Extended Concept of Voxel Neighborhoods for Correct Thinning in Mesh Segmentation, März 2005, Chemnitz

CSR-05-03 Wolfgang Rehm (Ed.), Kommunikation in Clusterrechnern und Clusterverbundsystemen, Tagungsband zum 1. Workshop, Dezember 2005, Chemnitz

CSR-05-04 Andreas Goerdt, Higher type recursive program schemes and the nested pushdown automaton, Dezember 2005, Chemnitz

CSR-05-05 Amin Coja-Oghlan, Andreas Goerdt, André Lanka, Spectral Partitioning of Random Graphs with Given Expected Degrees, Dezember 2005, Chemnitz

CSR-06-01 Wassil Dimitrow, Mathias Sporer, Wolfram Hardt, UML basierte Zeitmodellierung für eingebettete Echtzeitsysteme, Februar 2006, Chemnitz

CSR-06-02 Mario Lorenz, Guido Brunnett, Optimized Visualization for Tiled Displays, März 2006, Chemnitz

CSR-06-03 D. Beer, S. Höhne, R. Kunis, G. Rünger, M. Voigt, RAfEG - Eine Open Source basierte Architektur für die Abarbeitung von Verwaltungsprozessen im E-Government, April 2006, Chemnitz

CSR-06-04 Michael Kämpf, Probleme der Tourenbildung, Mai 2006, Chemnitz

CSR-06-06 Torsten Hoefler, Mirko Reinhardt, Torsten Mehlan, Frank Mietke, Wolfgang Rehm, Low Overhead Ethernet Communication for Open MPI on Linux Clusters, Juli 2006, Chemnitz

CSR-06-07 Karsten Hilbert, Guido Brunnett, A Texture-Based Appearance Preserving Level of Detail Algorithm for Real-time Rendering of High Quality Images, August 2006, Chemnitz

Chemnitzer Informatik-Berichte

CSR-06-08 David Brunner, Guido Brunnett, Robin Strand, A High-Perpormance Parallel Thinning Approach Using a Non-Cubic Grid Structure, September 2006, Chemnitz

CSR-06-09 El-Ashry, Peter Köchel, Sebastian Schüler, On Models and Solutions for the Allocation of Transportation Resources in Hub-and-Spoke Systems, September 2006, Chemnitz

CSR-06-10 Raphael Kunis, Gudula Rünger, Michael Schwind, Dokumentenmanagement für Verwaltungsvorgänge im E-Government, Oktober 2006, Chemnitz

CSR-06-11 Daniel Beer, Jörg Dümmler, Gudula Rünger, Transformation ereignisgesteuerter Prozeßketten in Workflowbeschreibungen im XPDL-Format, Oktober 2006, Chemnitz

CSR-07-01 David Brunner, Guido Brunnett, High Quality Force Field Approximation in Linear Time and its Application to Skeletonization, April 2007, Chemnitz

CSR-07-02 Torsten Hoefler, Torsten Mehlan, Wolfgang Rehm (Eds.), Kommunikation in Clusterrechnern und Clusterverbundsystemen, Tagungsband zum 2. Workshop, Februar 2007, Chemnitz

CSR-07-03 Matthias Vodel, Mirko Caspar, Wolfram Hardt, Energy-Balanced Cooperative Routing Approach for Radio Standard Spanning Mobile Ad Hoc Networks, Oktober 2007, Chemnitz

CSR-07-04 Matthias Vodel, Mirko Caspar, Wolfram Hardt, A Concept for Radio Standard Spanning Communication in Mobile Ad Hoc Networks, Oktober 2007, Chemnitz

CSR-07-05 Raphael Kunis, Gudula Rünger, RAfEG: Referenz-Systemarchitektur und prototypische Umsetzung - Ausschnitt aus dem Abschlussbericht zum Projekt "Referenzarchitektur für E-Government" (RAfEG) -, Dezember 2007, Chemnitz

CSR-08-01 Johannes Steinmüller, Holger Langner, Marc Ritter, Jens Zeidler (Hrsg.), 15 Jahre Künstliche Intelligenz an der TU Chemnitz, April 2008, Chemnitz

CSR-08-02 Petr Kroha, José Emilio Labra Gayo, Using Semantic Web Technology in Requirements Specifications, November 2008, Chemnitz

CSR-09-01 Amin Coja-Oghlan, Andreas Goerdt, André Lanka, Spectral Partitioning of Random Graphs with Given Expected Degrees - Detailed Version, Januar 2009, Chemnitz

CSR-09-02 Enrico Kienel, Guido Brunnett, GPU-Accelerated Contour Extraction on Large Images Using Snakes, Februar 2009, Chemnitz

CSR-09-03 Peter Köchel, Simulation Optimisation: Approaches, Examples, and Experiences, März 2009, Chemnitz

Chemnitzer Informatik-Berichte

ISSN 0947-5125

Herausgeber: Fakultät für Informatik, TU Chemnitz
Straße der Nationen 62, D-09111 Chemnitz