International Journal of Mechanical, Industrial and Aerospace Sciences
ISSN: 2517-9950
Vol:6, No:7, 2012

Constrained Particle Swarm Optimization of Supply Chains

András Király, Tamás Varga, János Abonyi

***Abstract*—**Since supply chains highly impact the financial performance of companies, it is important to optimize and analyze their Key Performance Indicators (KPI). The synergistic combination of Particle Swarm Optimization (PSO) and Monte Carlo simulation is applied to determine the optimal reorder point of warehouses in supply chains. The goal of the optimization is the minimization of the objective function calculated as the linear combination of holding and order costs. The required values of service levels of the warehouses represent non-linear constraints in the PSO. The results illustrate that the developed stochastic simulator and optimization tool is flexible enough to handle complex situations.

***Keywords*—**stochastic processes, empirical distributions, Monte Carlo simulation, PSO, supply chain management

I. Introduction

THE determination of safety stock in an inventory model is one of the key tasks of supply chain management. Miranda and Garrido include safety stock in the inventory model in [12]. Authors in [4] give a model for positioning safety stock in a supply chain subject to non-stationary demand and show how to extend their former model to find the optimal placement safety stocks under constant service time (CST) policy. Prékopa in [16] gives an improved model for the so called Hungarian inventory control model to find the minimal safety stock level that ensures the continuous production, without disruption.

The bullwhip effect is an important phenomenon in supply chains. Authors in [10] show how a supply chain can be modeled and analyzed by colored petri nets (CPN) and CPN tools and they evaluate the bullwhip effect, the surplus of inventory goods, etc. using the beer game as demonstration. More recent research can be found in [1], which shows that an order policy applied to a serial single-product supply chain with four echelons can reduce or amplify the bullwhip effect and inventory oscillation. Miranda et al. investigate the modeling of a two echelon supply chain system and optimization in two steps [15], while a massive multi-echelon inventory model is presented by Seo [19], where an order risk policy for general multi-echelon system is given, which minimizes the system operation cost. A really complex system is examined in [20], where it is necessary to apply some clustering for similar items, because detailed analysis could become impossible considering each item individually. The stability of the supply chain is also an intensively studied area.

The author are with the University of Pannonia, Department of Process Engineering, Veszprém, H-8200, Hungary (phone: +36-70-944-8910; e-mail: janos@abonyilab.com).

[14] shows that a linear supply chain can be stabilized by the anticipation of the own future inventory and by taking into account the inventories of other suppliers, and Vaughan in [21] presents a linear order point/lot size model that with its robustness can contribute to business process modeling.

Based on the previous review it is clear that most of the multi-echelon supply chain optimization and analysis are mainly based on analytical approach. Simulation however provides a very good alternative, because it can model real life situations with accuracy, more flexible in terms of input parameters and therefore it is more easy to use in decision support. The simulation results can be analyzed with various statistical methods and numerical optimization algorithms. To analyze complex, especially multi-echelon systems, multi-level simulation models can be used, where the results of optimized high level model feeds into the lower level more detailed models.

The simulation-based approach was published only in the last decade. Jung et al. [7] make a Monte Carlo based sampling from real data, and apply a simulation–optimization framework while looking for managing uncertainty. They use a gradient-based search algorithm, while authors in [8] discuss how to use simulation to describe a five-level inventory system, and optimize this model by genetic algorithm. Schwartz et al. [18] demonstrate the internal model control (IMC) and model predictive control (MPC) algorithms to manage inventory in uncertain production inventory and multi-echelon supply/demand networks. A complex instance of inventory model can be found in [5], where orders cross in time considering various distributions for the lead time. Sakaguchi in [17] investigates the dynamic inventory model in which demands are discrete and varying period by period

The aim of our research is to create a Monte-Carlo simulator which uses probability distributions based on material usage data posted in the logistic module of an enterprise resource planning (ERP) system. The main objective of this development was to build a simulator that can use simple building blocks to construct models of complex supply chain networks. Supply chains processes can be simulated using these modular models, where parameters of Key Performance Indicators are analyzed by sensitivity analysis. The developed SIMWARE simulator can be used as a verification tool to analyze and evaluate inventory control strategies. The simulation of “actual” inventory controlling strategies provides the most important key performance indicators KPI-s of these strategies. On the other hand this simulator can be used for optimization to determine the optimal values of the key inventory control parameters.

The proposed SIMWARE software provides a framework to analyze the cost structure and optimize inventory control parameters based on cost objectives.

With this tool we have minimized the inventory holding cost by changing the parameters of the reordering strategy while keeping the service level at the required value. The simulation of "actual" inventory controlling strategies provides the most important KPI-s of these strategies. On the other hand we can use the simulator as part of optimization and determine the optimal values of the key inventory control parameters. We are in the process to finalize the costing model therefore we used a simple cost function at this point. We have minimized the inventory holding cost by changing the parameters of our operational space while keeping the service level at the required value.

In the last decades, optimization was featured in almost all aspects of human civilization, thus it has truly become an indispensable method. In some aspects, even a local optima can highly improve the efficiency or reduce the expenses, however, most companies want to keep their operational costs as low as possible, i.e. on global minimum. Problems where solutions must satisfy a set of constraints are known as constrained optimization problems. In inventory control theory, one of the most important and most strict constraints is the service level, i.e. the portion of satisfied demands from all customer needs. The particle swarm optimization algorithm has been successfully applied to a wide set of complex problems, like data mining [29], software testing [30], nonlinear mapping [31], function minimization [32] or neural network training [33] and in the last decade, constrained optimization using PSO got a bigger attention [34,35,36].

There exist some well-known conditions under which the basic PSO algorithm exhibits poor convergence characteristics [28]. However, only a few studies have considered the hybridization of PSO, especially making use of gradient information directly within PSO. Notable ones are HGPSO [25] and GTPSO [26], which use the gradient descent algorithm, and FR-PSO [27], which applies the Flecher-Reeves method. As it will be demonstrated in the following sections, combining these two methods appropriately, the efficiency of the optimization using PSO can be considerably improved.

The structure of the paper is the following: Section II is a general introduction to the problem describing the multi-echelon supply chain and the relevant cost structure and the proposed flexible modeling tool to build complex multi-echelon supply chain models using simple, easy to understand modules. Section III introduces the proposed optimization algorithm. Section IV represents the main results through a case study, while section V concludes our work.

II. Stochastic Multi-Echelon Supply chain model

A. Inventory model of a single warehouse

The modular model of the supply chain is based on the following classic model of inventory control. This session gives a summary of the most important parameters of this model. In Figure 1, *Q* is the theoretical demand over cycle time *T* and this is the *Order Quantity*; *R* is the *Reorder point*, which is the maximum demand can be satisfied during the replenishment lead time (*L*). The *Cycle time* (*T*) is the time between two purchase orders.

The *Order Quantity* is *Q*, where $Q = \bar{d} \cdot T$. This is the ordered quantity in a purchase order, and *Q* is equal to the *Expected demand* and the Maximum *stock level*. *Maximum stock level* is the stock level necessary to cover the *Expected demand* in period *T;* therefore it has to be the quantity we order. *Lead time*(*L*) is the time between the Purchase order and the goods receipt. $\bar{d}_L$ denotes the average demand during the replenishment lead time. $\bar{d}_L = \bar{d} \cdot L$, where $\bar{d}$ is the daily average demand. Using the same logic, $\bar{d}_L$ is a special case; it yields consumption if the service level is 100%. We will use $\bar{d}_L$ to denote the consumption during the paper. *Reorder point* is the stock level when the next purchase order has to be issued. It is used for materials where the inventory control is based on actual stock levels.

S is the *Safety stock*; this is needed if the demand is higher than the expected (line *d*). In an ideal case *R* equals to total of safety stock and average demand over lead time: $R = \bar{d}_L + S$, where *S* is the *Safety stock* which is defined to cover the stochastic demand changes. For a given *Service Level* this is the maximum demand can be satisfied over the Lead time.

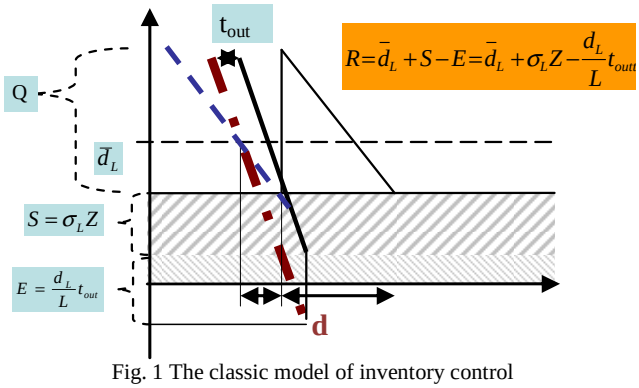


Fig. 1 The classic model of inventory control

Assuming constant demand pattern over the cycle time, Average Stock (*K*) can be calculated as a weighted average of stock levels over the cycle time:

$$K = \frac{Q}{2} + S \tag{1}$$

Service Level (*SL*) is the ratio of the satisfied and the total demand (in general this is the mean of a probability distribution), or in other words it is the difference between the 100% and the ration of unsatisfied demand:

$$SL = 100 - 100\frac{(d_L - R)}{Q} \tag{2}$$

We assume that all demand is satisfied from stock until stock exists. When we reach stock level *R* the demand over the lead time ($\bar{d}_L$) will be satisfied up to *R*. Consequently if $\bar{d}_L < R$, we are getting a stock out situation and there will be unsatisfied demand therefore the service level will be lower than 100%. $\bar{d}_L$ is not known and it is a random variable. The probability of a certain demand level is $P(\bar{d}_L)$. Based on this, the service level is formed as shown in the next equation:

$$SL = 100 - 100\frac{\int_{d_L}^{d_{\max}} P(d_L)(d_L - R)d_L}{Q} \quad (3)$$

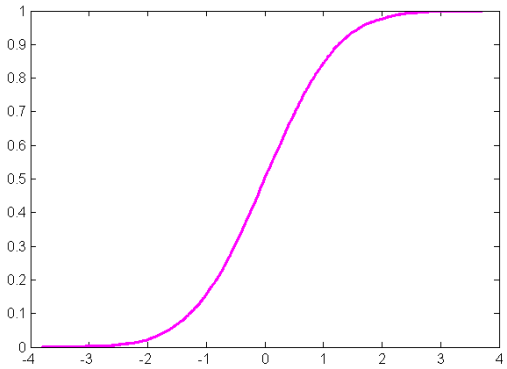
where $\bar{d}_L$ is continuous random variable, and $\bar{d}_{max}$ is the maximum demand over Lead time.

Based on our experience in analyzing actual supply chain systems we discovered that the probability functions of material flow and demand are different from the theoretical functions (see Figure 2 that shows the distribution function of an actual material consumption compared to the normal distribution used in most of the analytical methodologies). This difference makes difference between the theoretical (calculated) and the actual inventory movements, therefore it makes sense using a stochastic simulation approach based on "empirical" distribution functions.

Inventory movements can be modeled much better using stochastic differential equations than modeling based on the theoretical assumption that movements are following normal distribution. We propose the following model:

$$x_{L_{i+1}} = x_{L_i} - W_i + u(x, R, t_u) \quad (4)$$

Where x_i is stock level on the i^{th} week, W_i is a stochastic process to represent consumption. This stochastic process is based on the empirical cumulative distribution function we described in the previous section. u is the quantity of material received on week i, based on purchase orders. Purchase orders are calculated based on the actual inventory level (x), and the replenishment lead-time (t_u).



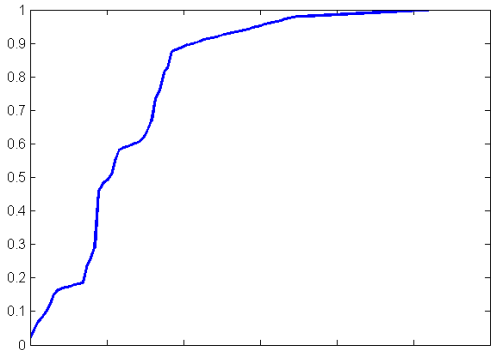


Fig. 2 The theoretical cumulative distribution function (top) and the actual cumulative distribution function for a raw material based on its consumption data (bottom)

B. Multi-echelon warehouse model

The main objective of the presented development is to build a simulator that can utilize the previously proposed building blocks to construct models of complex multi-echelon supply chain networks. In the following demonstrated example a simulator to analyze a system with two connected warehouses is presented. The following diagram shows the supply chain, i.e. the structure of the analyzed 2-level system.

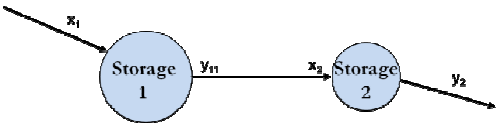


Fig. 3 The analyzed 2-level system

Where the objective function is:

$$f(z) = mean(h_1) + 1.3 * mean(h_2), \quad (5)$$

i.e. the holding cost in the second Warehouse is 30 percent higher than in the first Warehouse.

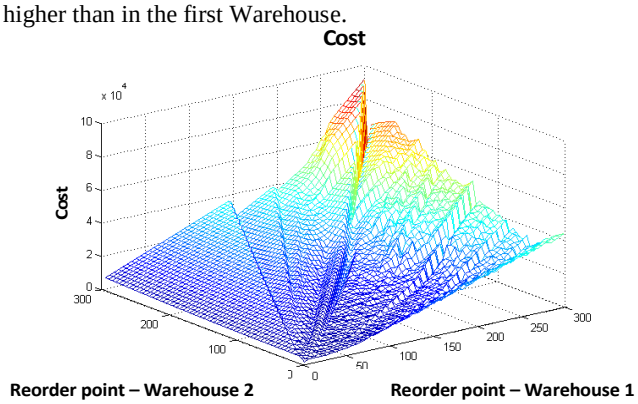


Fig. 4 The values of the objective function for the 2-level system

In Figure 4, the values of the objective function (i.e. cost) is presented as a function of the reorder point of the two Warehouses. Figure 5 shows the service level of Warehouse 1 in the 2-level system. The constraint for the service levels is 95% in this case.

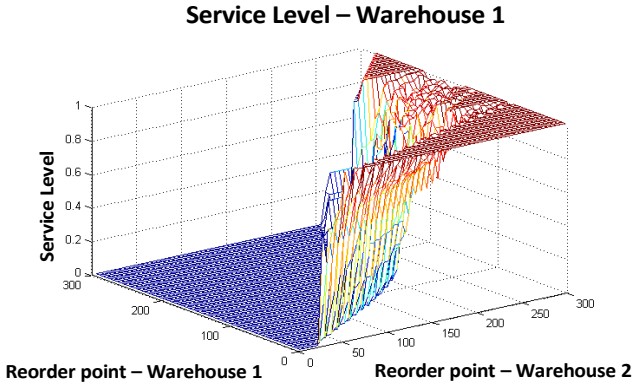


Fig. 5 The values of service level for Warehouse 1

The simulator is capable to optimize the two warehouses in the same time and calculate the optimum for the supply chain as a whole. The required values of service levels of the warehouses represent non-linear constraints, hence a nonlinear optimization algorithm developed to solve this complex optimization problem.

III. The Improved PSO Algorithm

There are two popular swarm inspired methods in computational intelligence areas: Ant colony optimization (ACO) and PSO. ACO was inspired by the behaviors of ants and has many successful applications in discrete optimization problems. The particle swarm concept originated as a simulation of simplified social system. The original intent was to graphically simulate the choreography of bird of a bird block or fish school. However, it was found that particle swarm model can be used as an optimizer. Suppose the following scenario: a group of birds are randomly searching food in an area. There is only one piece of food in the area being searched. All the birds do not know where the food is. But they know how far the food is in each iteration. So what's the best strategy to find the food? The effective one is to follow the bird which is nearest to the food.

PSO is based on this scheme. This stochastic optimization technique has been developed by Eberhart and Kennedy in 1995 [22]. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles. All of particles have fitness values which are evaluated by the fitness function to be optimized, and have velocities which direct to the flying of the particles.

PSO is initialized with a group of random particles (solutions) and then searches for optima by updating generations. In every iteration, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far. (The fitness value is also stored.) This value is called *pbest*. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population. This best value is a global best and called *gbest*. When a particle takes part of the population as its topological neighbors, the best value is a local best and is called *lbest*.

$$v_j(k+1) = w \cdot v_j(k) + c_1 \cdot rand() \cdot \left(x_{pbest} - x_j(k)\right) + c_2 \cdot rand() \cdot \left(x_{gbest} - x_j(k)\right) \quad (6)$$

$$x_j(k+1) = x_j(k) + v_j(k+1) \cdot dt \quad (7)$$

Where *v* is the particle velocity, *pbest* and *gbest* are defined as stated before, *rand()*is a random number between [0,1], *c1*, *c2* are learning factors usually *c1=c2=2*. Code 1. shows the pseudo code of the PSO algorithm.

Code 1 The pseudo code of the PSO algorithm

```
procedure PSO; {
    Initialize particles;
    while (not terminate) do {
      for each particle {
        Calculate fitness value;
        if fitness <pBest than  pBest = fitness;
      }
      Choose the best particle as the gBest;
      for each particle {
        Calculate particle velocity;
        Update particle position;
      }
    }
}
```

The role of the, *w*, inertia weight in Eq. (6), is considered critical for the convergence behavior of PSO. The inertia weight is employed to control the impact of the previous history of velocities on the current one. Accordingly, the parameter regulates the trade–off between the global and local exploration abilities of the swarm. A large inertia weight facilitates global exploration (searching new areas) while a small one tends to facilitate local exploration, i.e. fine–tuning the current search area.

PSO shares many similarities with evolutionary computation techniques, e.g. with evolutionary algorithms (EAs). Both algorithms start with a group of a randomly generated population, both have fitness values to evaluate the population. Both update the population and search for the optimum with random techniques. Both systems do not guarantee success. The main difference between these algorithms is that PSO does not have genetic operators like crossover and mutation. Particles update themselves with the internal velocity. They also have memory, which is important to the algorithm.

Compared with evolutionary algorithms, the information sharing mechanism in PSO is significantly different. In EAs, chromosomes share information with each other. So the whole population moves like a one group towards an optimal area. In PSO, only *gBest* (or *lBest*) gives out the information to others. It is a one-way information sharing mechanism, the evolution only looks for the best solution. Compared with EAs, all the particles tend to converge to the best solution quickly even in the local version in most cases. Compared to EA, the advantages of PSO are that PSO is easy to implement and

there are few parameters to adjust. Hence, PSO has been successfully applied in many areas: function optimization, artificial neural network training, fuzzy system control, and other areas where GA can be applied.

A. The proposed algorithm

As we saw in section I., the basic PSO algorithm exhibits poor convergence characteristics under some specific conditions. We gave a small overview also about the previous gradient based methods, and in this section we will demonstrate a novel way, how the particle swarm optimization (PSO) technique can be improved with the calculation of the gradient of the applied objective function. There are some well documented algorithms in the literature to boost the convergence of the basic PSO algorithm. Victoire et al. developed a hybrid PSO to solve the economic dispatch program. They combined PSO with Sequential Quadratic Programming to search for the gradient of the objective function. A very similar algorithm is introduced by Noel, in which quasi Newton-Raphson (QNR) algorithm is applied to calculate the gradient [23]. The QNR algorithm optimizes by locally fitting a quadratic surface and finding the minimum of that quadratic surface.

Our aim is to develop a novel PSO algorithm which is able to consider linear and non-linear constraints and it calculates the gradient of the objective function to improve the affectivity.

PSO is initialized with a group of random particles (solutions) and then searches for optima by updating generations. In every generation, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far. This value is called *pbest*. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population. This best value is a global best and called *gbest*. When a particle takes part of the population as its topological neighbors, the best value is a local best and is called *lbest*. Our vision is to apply the gradient of the objective function in every generation to control the movements of the particles. Therefore, the equation which is applied to calculate the velocity of the particles is modified:

$$\begin{aligned} v_j(k+1) &= w \cdot v_j(k) + \\ & c_1 \cdot rand() \cdot \left(x_{pbest} - x_j(k)\right) + \\ & c_2 \cdot rand() \cdot \left(x_{gbest} - x_j(k)\right) + \\ & + c_3 \cdot grad\left(f(x)\right) \end{aligned} \tag{8}$$

$$x_j(k+1) = x_j(k) + v_j(k+1) \cdot dt \tag{9}$$

Where $grad(f(x))$ represents the partial derivatives of the objective function, and c_3 is the weight for the gradient term. In Noel's work a uniformly distributed random value is applied as c_3 from the interval of $[0,0.5]^2$. Since the negative gradient always points in the direction of steepest decrease in the function, the nearest local minimum will be reached eventually. Since the gradient is zero at a local minimum, smaller steps will automatically be taken when a minimum is approached. Also, movement in a direction other than the direction defined by the negative gradient will result in a smaller decrease in the value of the cost function.

B. Illustrative example

In the following a simple illustrative example is presented to demonstrate the efficiency of the proposed algorithm.

The aim of the optimization is to minimize two objective functions with two variables:

$$f(x,y) = x^2 + y^2 + xy + 5 \tag{10}$$

$$f(x,y) = \sin(120/x) + \cos(60/y) \tag{11}$$

The gradients of these functions can be analytically calculated:

$$\frac{\partial f(x,y)}{\partial x} = 2x + y \tag{12}$$

$$\frac{\partial f(x,y)}{\partial y} = 2y + x \tag{13}$$

and

$$\frac{\partial f(x,y)}{\partial x} = -\frac{120 \cdot \cos(120/x)}{x^2} \tag{14}$$

$$\frac{\partial f(x,y)}{\partial y} = \frac{60 \cdot \sin(60/y)}{y^2} \tag{15}$$

The determined gradients are applied to increase the convergence of the search. The shape of the two analyzed functions can be seen in Figure 5. Both of the functions have a global minimum (at 5 and -2).

In Table I, the analysis of c_3 is summarized in these two cases. As can be seen in both cases the necessary number of generations is lower if the gradient is applied to control the movement of particles than in case the value of c_3 is 0. It means that the application of gradient can increase the convergence of PSO algorithm.The proper value of c_3 is close in these two investigations. However, to determine a universally applicable value more objective functions must be analyzed and many evaluations must be performed. In that case we have a proper value for c_3 the PSO algorithm can be further improved with the integration of Monte Carlo simulation to numerically determine the gradient.

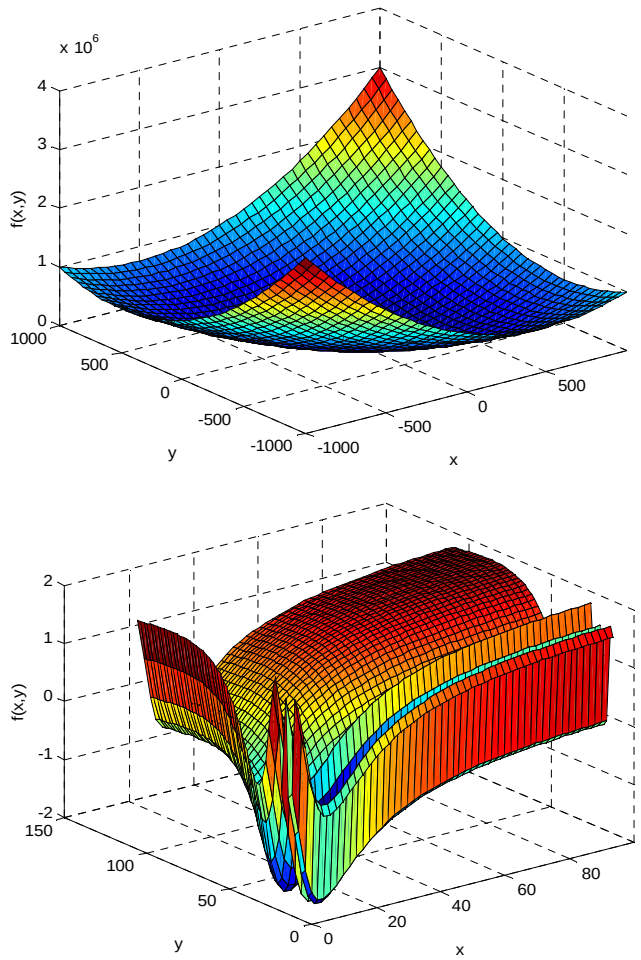


Fig. 6 Shape of the analyzed objective functions

IV. Results

The developed simulator and optimization algorithm can be used to optimize the two warehouses in the same time and calculate the optimum for the supply chain as a whole. Authors developed a new component-based MATLAB simulator, as well as a novel PSO algorithm. The algorithm is based on an existing implementation form MATLAB Central [37]. The reason we choose this Toolbox is that it can handle linear and Non-linear constraints also. It handles nonlinear inequality constraints in the form c(x) <= 0 using 'soft' or 'penalize' boundaries. The penalization is like "soft" boundaries, except that some kind of penalty value must be calculated from the degree of each constraint violation.

MC simulation is applied to describe the stochastic behavior of the process. The investigated period is 28 weeks. The service levels of both of the warehouses are determined. Ten MC simulations are evaluated to simulate different situations and the average of the simulated results is used to calculate the value of the applied objective function at given reorder point. PSO is applied to modify the reorder point due to the value of the objective function and finally to find the global optima.

Before the optimization the reorder points of the two warehouses are 160 and 60 units.

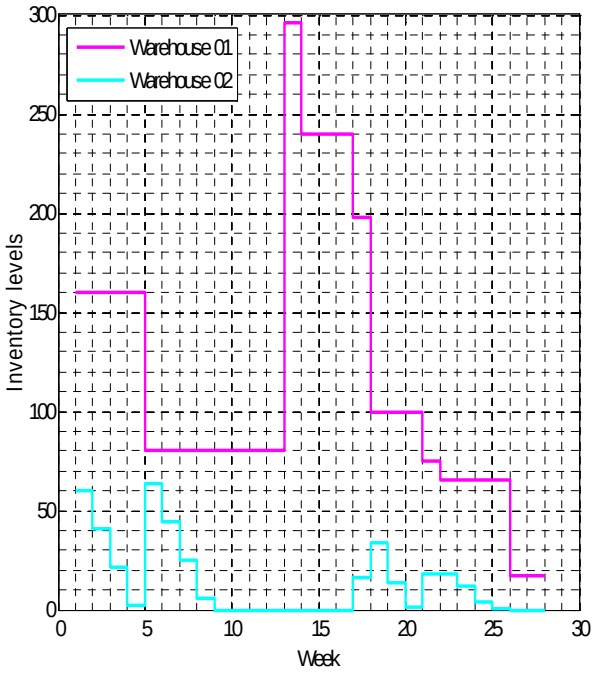
At the initial reorder points the actual service levels are below the desired values (0.95 and 0.90) in both of the warehouses (0.76 and 0.63). After the optimization process the reorder points is changed to 200 and 71.708. Due to this modification the service levels are much better than at the initial state (0.97 and 0.89).

The average inventory levels before and after optimization can be seen in Figure 7. It can be seen that before optimization the inventory of the Warehouse 02 is depleted between the 9th and 17th weeks and after 25th weeks. Due to the optimization the inventory in the second warehouse is not empty in these crucial periods.

Figure 8 shows only the service level for the first warehouse, but in the optimization problem both service levels were taken into consideration. The optimal solution is highlighted with the green square. It satisfies the 95% constraints and ensures the minimal holding cost in the warehouses.

TABLE I
The Affect Of C_3 On The Convergence Of PSO

c_3	0	0.01	0.02	0.05	0.06	0.07	0.08	0.09	0.1
Generations (case eq. 3)	127	134	122	136	116	125	138	135	142
Generations (case eq. 4)	114	93	99	86	90	68	103	110	97



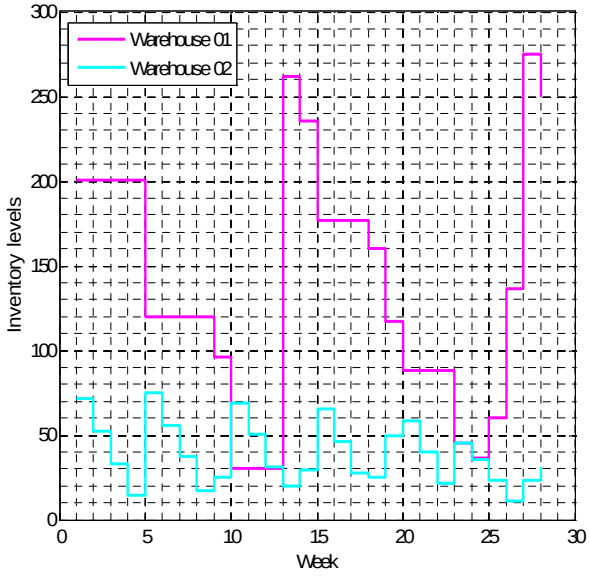


Fig. 7 The inventory levels in the two-level system before optimization (a: left hand side) and after optimization (b: right hand side)

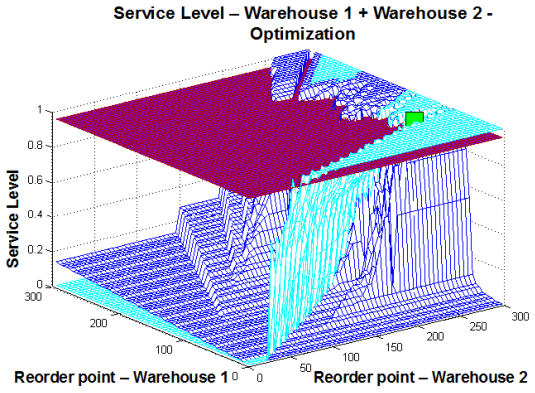


Fig. 8 Illustrative representation of the cost function and the constraint of the service level

V. Conclusion

Since supply chain performance impacts the financial performance companies, it is important to optimize and analyze their performances. For this purpose an interactive simulator, SIMWARE, capable to simulate complex multi-echelon supply chains based on simple configurable connection of building blocks has been developed. To support the optimization of supply chains as complex systems a Particle Swarm Optimization algorithm that utilizes gradients (sensitivities) was developed. The proposed method is applied in case of multi-echelon system built from two warehouses. We validated our solution by simulating four stochastic input variables. The results illustrate that the developed tool is flexible enough to handle complex situations and straightforward and simple enough to be used for decision support..

Acknowledgment

This work was supported by the European Union and financed by the European Social Fund in the frame of the TAMOP-4.2.1/B-09/1/KONV-2010-0003 project and EON Business Services Ltd (Hungary).

References

[1] G. Caloiero, F. Strozzi, J. M. Z. Comenges:A supply chain as a series of filters or amplifiers of the bullwhip effect, International Journal of Production Economics, 114 (2008) 2, pp. 631–645
[2] S. C. Graves, S. P. Willems: Optimizing strategic safety stock placement in supply chains, Manufacturing & Service Operations Management, 2 (2000) 1, pp. 68–83
[3] S. C. Graves, S. P. Willems: Supply chain design: safety stock placement and supply chain configuration. Handbooks in operations research and management science, 11 (2003) pp. 95–132
[4] S. C. Graves, S. P. Willems: Strategic inventory placement in supply chains: Nonstationary demand, Manufacturing & Service Operations Management, 10 (2008) 2, pp. 278–287
[5] J. C. Hayya, U. Bagchi, J. G. Kim, D. Sun: On static stochastic order crossover, International Journal of Production Economics,114 (2008) 1, pp. 404–413
[6] K. F. Simpson Jr: In-process inventories, Operations Research, (1958), pp. 863–873
[7] J. Y. Jung, G. Blau, J. F. Pekny, G. V. Reklaitis, D. Eversdyk: A simulation based optimization approach to supply chain management under demand uncertainty, Computers & chemical engineering,28 (2004) 10, pp. 2087–2106
[8] P. Köchel and U. Nieländer: Simulation-based optimisation of multi-echelon inventory systems. International Journal of Production Economics, 93 (2005), 505–513
[9] A. H. L. Lau, H. S. Lau: A comparison of different methods for estimating the average inventory level in a (q, r) system with backorders, International Journal of Production Economics, 79 (2003) 3, pp. 303–316
[10] D. Makajic-Nikolic, B. Panic, M. Vujosevic: Bullwhip effect and supply chain modeling and analysis using cpn tools, Fifth Workshop and Tutorial on Practical Use of Colored Petri Nets and the CPN Tools, Citeseer,(2004)
[11] H. Min, G. Zhou: Supply chain modelling: past, present and future, Computers & Industrial Engineering, 43 (2002) 1-2, pp. 231–249
[12] P. A. Miranda, R. A. Garrido: Incorporating inventory control decisions into a strategic distribution network design model with stochastic demand, Transportation Research Part E: Logistics and Transportation Review, 40 (2004), pp. 183–207
[13] E. P. Musalem, R. Dekker: Controlling inventories in a supply chain: a case study, International Journal of Production Economics, 93 (2005) pp. 179–188
[14] T. Nagatani, D. Helbing: Stability analysis and stabilization strategies for linear supply chains, Physica A: Statistical and Theoretical Physics, 335 (2004) 3-4 pp. 644–660
[15] P. A. Miranda, R. A. Garrido: Inventory service-level optimization within distribution network design problem, International Journal of Production Economics, 122 (2009) 1, pp. 276–285
[16] A. Prékopa: On the Hungarian inventory control model, European journal of operational research, 171 (2006) 3, pp. 894–914
[17] M. Sakaguchi: Inventory model for an inventory system with time varying demand rate, International Journal of Production Economics, 122 (2009) 1, pp. 269–275
[18] J. D. Schwartz, W. Wang, D. E. Rivera: Simulation-based optimization of process control policies for inventory management in supply chains, Automatica, 42 (2006) 8, pp. 1311–1320
[19] Y. Seo: Controlling general multi-echelon distribution supply chains with improved reorder decision policy utilizing real-time shared stock information, Computers & Industrial Engineering, 51 (2006) 2, pp. 229–246
[20] M. Srinivasan, Y. B. Moon: A comprehensive clustering algorithm for strategic analysis of supply chain networks, Computers & industrial engineering, 36 (1999) 3, pp. 615–633
[21] T. S. Vaughan: Lot size effects on process lead time, lead time demand, and safety stock, International Journal of Production Economics, 100 (2006) 1, pp. 1–9

[22] J. Kennedy, R.C.Eberhart, Particle swarm optimization, Proceedings of IEEE International Conference on Neural Networks, 1942-1948, 1995 (1)
[23] T.A.A. Victoire, A.E. Jeyakumar, Hybrid PSO-SQP for economic dispatch with valve-pointeffect, Electric Power Systems Research, 71, 51-59, 2004 (2)
[24] M.M Noel, new gradient based particle swarm optimization algorithm for accurate computation of global minimum, 12, 353-359, 2012
[25] M.M. Noel and T.C. Jannett, Simulation of a new hybrid particle swarm optimization algorithm. In Proceedings of the Thirty-Sixth Southeastern Symposium on System Theory, pp. 150-153, 2004
[26] R. Zhang, W. Zhang and X. Zhanf, A new hybrid gradient-based particle swarm optimization algorithm and its applications to control of polarization mode dispersion in optical fiber communication systems. *The 2009 International Joint Conference on Computational Sciences and Optimization*, pp. 1031-1033, 2009.
[27] B. Borowska and S. Nadolski, Particle swarm optimization: the gradient correction. *Journal of Applied Computer Science*, 17(2):7-15, 2009.
[28] F. Van den Bergh, An Analysis of Particle Swarm Optimizers. *Phd thesis*, University of Pretoria, South Africa, 2002.
[29] T. Sousa, A. Silva and A. Neves, Particle swarm based data mining algorithms for classification tasks. *Parallel Computing*, 30:767-783, 2004.
[30] A. Windisch, S. Wappler and J. Wegener, Applying particle swarm optimization to software testing. *In Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pp. 1121-1128, 2007.
[31] A. I. Edwards, A.P. Engelbrecht and N. Franken, Nonlinear mapping using particle swarm optimisation. *The 2005 IEEE Congress on Ecvolutionary Computation*, 1:306-313, 2005.
[32] J. Kennedy and R. C. Eberhart, Particle swarm optimization. In *Proceedings of the IEEE Internation Conference on Neural Networks*, vol 4, pp. 1942-1948, 1995.
[33] A. P. Engelbrecht and A. Ismail, Training product unit neural networks. *Stability and Control: Theory and Applications*, 2:59-74, 1999
[34] K.E. Parsopoulos and M.N. Vrahatis, Particle swarm optimization method for constrained optimization problems, *Intelligent Technologies--Theory and Application: New Trends in Intelligent Technologies*, vol. 76, pp. 214-220, 2002.
[35] X. Hu and R. Eberhart, Solving constrained nonlinear optimization problems with particle swarm optimization, *Proceedings of the sixth world multiconference on systemics, cybernetics and informatics*, vol. 5, pp. 203-206, 2002.
[36] T. Wimalajeewa and S.K. Jayaweera, Optimal power scheduling for correlated data fusion in wireless sensor networks via constrained PSO, *IEEE Transactions on Wireless Communications*, 7(9), 3068-3618, 2008.
[37] S. Chen, Another Particle Swarm Toolbox, *MATLAB Central*, 2010, http://www.mathworks.com/matlabcentral/fileexchange/25986

András Király received the MSc degree in information technology in 2010 from the University of Pannonia, Hungary. Currently, he is a PhD student in the Doctoral School in Chemical Engineering and Material Sciences at University of Pannonia. In the period of 2011-2012 he was employed at the Finnish Microarray and Sequencing Centre, Turku, Finland. His research interests include data-mining and evolutionary computation.

Tamás Varga received the MEng and PhD degrees in chemical engineering in 2006 and 2010 from the University of Pannonia, Hungary, respectively. Currently, he is a senior lecturer at the Department of Process Engineering at the University of Pannonia. His research interests include computer aided process engineering, computational fluid dynamics and process safety.

Janos Abonyi received the MEng and PhD degrees in chemical engineering in 1997 and 2000 from the University of Veszprem, Hungary, respectively. In 2008, he earned his Habilitation in the field of Process Engineering, and the DSc degree from the Hungarian Academy of Sciences in 2011 . Currently, he is a full professor at the Department of Process Engineering at the University of Pannonia. In the period of 1999-2000 he was employed at the Control Laboratory of the Delft University of Technology (in the Netherlands). Dr. Abonyi has co-authored more than 100 journal papers and chapters in books and has published two research monographs and one Hungarian textbook about data mining. His research interests include process engineering, quality engineering, data-mining and business process redesign.