April, 2008

Theoretical Framework of Multi-Objective Simulation-Based Genetic Algorithm for Supply Chain Cyclic Planning and Optimisation

Liana Napalkova
Galina Merkuryeva

Theoretical Framework of Multi-Objective Simulation-Based Genetic Algorithm for Supply Chain Cyclic Planning and Optimisation

Liana Napalkova and Galina Merkuryeva
Riga Technical University
1 Kalku Street, LV-1658, Riga, Latvia,
E-mail: liana@itl.rtu.lv, gm@itl.rtu.lv

Abstract

This paper develops a multi-objective simulation-based genetic algorithm (MOSGA) for multi-echelon supply chain cyclic planning and optimisation. The problem involves a search in high dimensional space with different ranges for decision variables scales, multiple objectives and problem specific constraints, such as power-of-two and nested/inverted-nested planning policies.

In order to find the optimal solution, different parameters of genetic algorithm including the population sizing, crossover and mutation probabilities, selection and reproduction strategies and convergence criteria are investigated. For finding approximations of the Pareto optimal set, the non-dominated sorting approach is used.

1. Introduction

For the last years, there has been increasing attention placed on the performance, design, and analysis of multi-echelon supply chains [1]. There are two different approaches used to manage supply chains: *single-echelon* and *multi-echelon approaches*.

The single-echelon approach, which includes continuous review, periodic review, single-unit decomposition, Wagner Within, Silver Meal techniques etc., splits multi-level supply chain into separate stages providing suboptimal solutions.

In contrast to it, the multi-echelon approach, which could be applied to non-cyclic and cyclic policies, considers managing all the echelons in a holistic way and, thus, optimises the global supply chain performance. Compared to non-cyclic policies, which are more preferable from theoretical point of view, *cyclic planning* in multi-echelon environment has more practical benefits, because it provides easy control, reduced administrative costs and safety stocks, and elimination of bullwhip effect [2]. The main idea of cyclic planning is to use cyclic schedules at each echelon and synchronize them with one another [3].

Multi-echelon cyclic planning and optimisation refers to the class of multi-objective optimization problems, which are usually characterized by a large search space of decision variables, conflicting and stochastic objectives etc. [4]. While there is no a single optimal solution for a number of conflicting objectives, the development of an algorithm, which gives a large number of alternative solutions lying near the Pareto-optimal frontier and tackles the variations of a response generated from the uncertainties in the decision variables and/or parameters, is of great practical value [5].

Different complex models based on mathematical inventory theory, such as mixed integer programming, dynamic programming, non-linear programming models etc., have been developed to define optimal cyclic policies. However, real-world supply chains cannot expect to get a solution by an analytical method.

The motivation for the current study is to propose a theoretical framework of the multi-objective genetic algorithm for solving this optimisation problem in accordance with research conducted within ECLIPS Specific Targeted Research Project of the European Commission "Extended Collaborative Integrated Life Cycle Supply Chain Planning System".

This paper is divided into eight sections. Section 2 describes multiple objectives and the optimisation problem statement. Section 3 contains requirements imposed on the development of genetic algorithm (GA) to solve the optimisation problem. Section 4 provides arguments in favour of the development of the multi-objective simulation-based GA. The main features of the used GA are discussed in sections 5.1-5.8. Section 6 presents a prototype of that algorithm. Conclusions and the acknowledgements are represented in Section 7 and Section 8, respectively.

2. Optimisation problem statement

The multi-objective simulation-optimisation problem can be symbolically represented in compact form as:

$$\begin{aligned}&\text{Minimize } \mathrm{E}[\boldsymbol{y}] = \mathrm{E}[F(\boldsymbol{x})] = \mathrm{E}[(f_1(\boldsymbol{x}),\ldots,f_n(\boldsymbol{x}))],\\ &\text{where} \quad \boldsymbol{x} = (x_1,\ldots x_m) \in X, \qquad (1)\\ &\qquad\qquad \boldsymbol{y} = (y_1,\ldots,y_n) \in Y,\end{aligned}$$

where $\mathrm{E}[\cdot]$ is a mathematical expectation, $\boldsymbol{x}$ is called a decision variable vector; $\boldsymbol{y}$ is called an objective vector; $x_1,\ldots x_m$ denote m decision variables; $y_1,\ldots,y_n$ denote n multiple objectives; X is a decision variable space; Y is an objective space.

Proceeding from (1), the solution of multi-objective optimisation problem is a vector of decision variables $\boldsymbol{x}$ that satisfies all feasible constraints and provides the best trade-off between multiple objectives. For finding trade-off solutions, it is necessary to use the principles of dominance and Pareto optimality [6,7].

The trade-off solution $\boldsymbol{x}\in X$ is said to be Pareto optimal with respect to X if and only if there is no $\boldsymbol{x}'\in X$, for which $F(\boldsymbol{x}') = (f_1(\boldsymbol{x}'),\ldots,f_n(\boldsymbol{x}'))$ dominates $F(\boldsymbol{x}) = (f_1(\boldsymbol{x}),\ldots,f_n(\boldsymbol{x}))$. At that, the set of objective vectors $F(\boldsymbol{x})$ corresponding to Pareto optimal vectors $\boldsymbol{x}\in\Omega$, where $\Omega\subseteq X$, is called Pareto-optimal frontier. Finding the whole Pareto-optimal frontier is a necessary condition for selecting trade-off solutions.

Regarding the problem of cyclic planning within multi-echelon supply chain environment we deal with two multiple objectives. The first one is to minimize the average total cost represented by sum of inventory holding, production and ordering costs in accordance with the following equation:

$$\text{Minimize } \mathrm{E}[TC] = \sum_{t=1}^{T}\Big(\sum_{j=1}^{J} CP_{j,t} + \sum_{i=1}^{I} CO_{i,t} + \sum_{i=1}^{I} CH_{i,t}\Big) \qquad (2)$$

where TC denotes the total cost, $CP_{i,t}$ denotes production cost in process j per period t, $CO_{i,t}$ is ordering cost at stock point i per period t, and $CH_{i,t}$ - inventory holding cost at stock point i per period t; I and J correspond to the number of stock points and processes, and T defines the number of periods in the planning horizon.

The second objective is to satisfy customer service requirements specified by the order fill rate.

$$\text{Maximize } \mathrm{E}[FR] = \sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{k=1}^{K} QC_{i,k,t} \Big/ \sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{k=1}^{K} D_{k,i,t} \qquad (3)$$

where $QC_{i,k,t}$ is a fraction of orders provided by stock point i to end-customer k in time period t, $D_{k,i,t}$ is actual demand of end-customer k to stock point i in time period t.

Controlled in optimisation experiments, this performance measure is introduced to avoid unconstrained minimization of the total cost.

The decision variables are replenishment cycles and order-up-to levels. Here, cycles are considered as independent variables, and order-up-to levels are re-calculated by using simplified analytical formulas described below.

To describe objective vector function (1), one could use traditional methods of transforming multiple objectives into a single objective through different scalarization and objective combination methods. The main strength of this approach is a computational efficiency and simple implementation. The weakness is the difficulty to determine a value of the weights that reflect a relative importance of each criterion. Moreover, it includes serious drawback in terms of appropriate representation of the real-world problem and the resulting solution qualities [11].

The proposed approach is to use Genetic Algorithm (GA) to guide the search towards Pareto optimal frontier. Compared to other algorithms, GAs are well suited for solving such problems as they are based on biological processes, which are inherently multi-objective.

3. Requirements the solution method

The main requirements imposed on the development of GA follow from analysis of multi-objective function behavior, a search space and specific constraints related to the investigated problem.

The multi-objective function is supposed to be:

- Non-linear and non-convex;
- Non-differentiable;
- Stochastic;
- Multi-modal.

Search space of decision variables is:

- Multi-dimensional;
- With different ranges for scales.

Specific constraints can be defined as:

- Use of power-of-two and nested/inverted nested inventory replenishment policies;
- Existence of feasible regions in a search space;
- Robustness of solutions.

Stochastic component of multi-objective function results in variations of objective response. Variations lead to uncertainties in decision variables. Moreover, the constraining relations between variables cause infeasible regions in search space. Because of these difficulties and the presence of local minimums, an analytical form and gradients of multi-objective function cannot be a priori derived. Therefore the most preferable technique is to develop GA in combination with the simulation model of multi-echelon supply chain to solve the investigated optimisation problem.

4. The proposed solution method

The simulation-based approach allows the multi-objective function: (i) to be an implicit function of decision variables of the system, (ii) to contain complex interactions between the system entities and (iii) to be stochastic in nature [9].

In general, different optimisation algorithms could utilize multi-echelon supply chain simulation models for estimating values of the above-described objective function. These methods refer to the so called Simulation-Optimisation approach.

In this paper, the choice in favour of GA has been dictated by the following reasons [10]:

- GA excellently deals with the random output of simulation experiments;
- GA can be designed to search for the entire set of Pareto optimal solutions in a single run, because it is based on evolving a population of solutions in parallel;
- GA does not make assumptions about the shape and mathematical properties of the Pareto frontier.
- GA is a heuristic search algorithm, which doesn't use gradients to find the search direction;
- GA is a global search method, which can be applied to functions with multiple local optima.

The successful performance of GA is dependent on many factors: the type and rate of crossover and mutation operators, population size, the encoding approach, etc. Currently, GA practitioners pick and adjust GA parameters empirically until they achieve adequate performance for a given problem.

The proposed Multi-Objective Simulation based Genetic Algorithm (MOSGA) is the result of analysis and a partial modifying properties of so called Non-dominated Sorting Genetic Algorithm II (NSGA-II) and NSGA-II described in [13,14]. This analysis refers to GA elements and parameters as follows:

1. Fitness assignment;
2. Encoding mechanism;
3. Initial population and fitness estimation through simulation;
4. Population sizing;
5. Crossover and mutation operators;
6. Selection strategy;
7. Reproduction strategy;
8. Termination criterion.

General scheme of MOSGA including their executive main operators is shown in Figure 1.

5. MOSGA description

The basic idea of MOSGA is to inherit properties of NSGA-II and adapt them to multi-echelon supply chain cyclic planning and optimisation task within the simulation-based environment.

5.1. Fitness assignment

In extending the ideas of GA to multi-objective cases, the first task is to accomplish fitness assignment and selection in order to guide the search towards Pareto optimal set.

At present, the majority of multi-objective GAs introduce the function of fitness assignment proposed by Goldberg [12]. The idea is to rank the population according to Pareto optimality. In the above mentioned NSGA algorithm, the first rank is assigned to the solutions that are non-dominated in the current population. After that, solutions are temporally removed from the population, and a new rank is assigned to the solutions non-dominated in the remaining part of the population. The process is continued until all the solutions in the population are ranked. However, this approach has (i) a high computational complexity of non-dominated sorting, (ii) lack of elitism and (iii) need for specifying the sharing parameter. An improved NSGA-II algorithm used in the paper has the features, such as (i) a *crowded comparison operator* ($\geq_n$) and (ii) a *crowding distance measure.* The first procedure guides the selection process at the various stages of the algorithm towards a uniformly spread out Pareto-optimal frontier, and the second procedure estimates the size of the largest *cuboid* enclosing the point without including any other point in the population. The advantage of Pareto ranking is its independence of any monotonic transformation of objective function. However, it should be taken into account that this kind

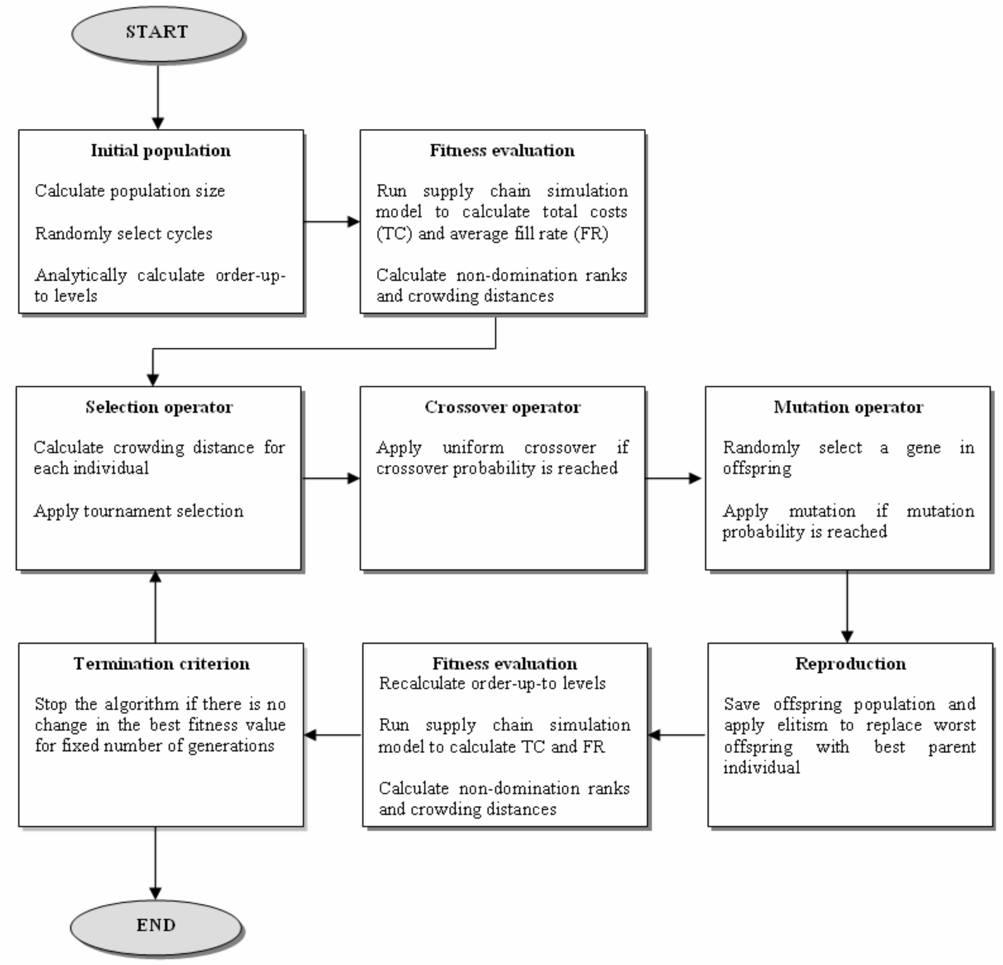

START
Initial population
Calculate population size
Randomly select cycles
Analytically calculate order-up-to levels
Fitness evaluation
Run supply chain simulation model to calculate total costs (TC) and average fill rate (FR)
Calculate non-domination ranks and crowding distances
Selection operator
Calculate crowding distance for each individual
Apply tournament selection
Crossover operator
Apply uniform crossover if crossover probability is reached
Mutation operator
Randomly select a gene in offspring
Apply mutation if mutation probability is reached
Reproduction
Save offspring population and apply elitism to replace worst offspring with best parent individual
Fitness evaluation
Recalculate order-up-to levels
Run supply chain simulation model to calculate TC and FR
Calculate non-domination ranks and crowding distances
Termination criterion
Stop the algorithm if there is no change in the best fitness value for fixed number of generations
END

Figure 1. Scheme of MOSGA algorithm

of fitness assignment may promote regions with higher density of solutions. In such regions, a given solution may dominate many others, even if the differences on objective values are very low.

5.2. Encoding mechanism

The encoding is aimed to produce a chromosome-like structure that will subsequently be used for breeding through the action of the various genetic operators. Binary encoding is the most widely used representation, where the fundamental elements are selected from the alphabet {0, 1}. There are a number of reasons why binary representation has dominated GA research including the Goldberg's *principle of minimal alphabets*, simple analysis of binary vectors and position independence.

In the proposed MOSGA algorithm the binary encoding is used because of benefits, which it provides to the problem representation. Note that multi-echelon cyclic planning introduces additional constraints to decision variables, such as power-of-two and nested/inverted nested policies. In a case of the binary encoding the replenishment cycles always satisfy a power-of-two policy.

The following two-step procedure is used in MOSGA to encode replenishment cycles to a 3-bit long string that is enough to define cycles search space:

1. Represent a cycle as a multiple of two, e.g. $8 = 2^3$;
2. Encode a power to a binary string, e.g. 3 => 011.

It could be noticed that the power '7' is the maximal one produced by a string '111'.

5.3. Initial population and fitness estimation through simulation

Initial population is a set of initial points used to start the search process. Ideally, it should be representative enough to cover the investigated search space. Usually, the initial population is generated randomly. MOSGA applies the following six-step procedure to produce initial solutions:

1. Generate randomly power-of-two cycles of each chromosome;
2. Sort cycles subject to nested (or inverted-nested) policy;
3. Calculate order-up-to levels for each chromosome using approximate analytical formulas [4];
4. Calculate average total costs and average fill rate of each chromosome by running a supply chain simulation model;
5. Define fitness values of chromosomes using a Pareto-ranking approach (rank 1 assigns the best solution);
6. Order chromosomes according to their fitness values in increasing sequence.

Step 1. In order to avoid any bias at the beginning of the evolutionary run, each cycle parameter $Cy \in [1,128]$ in the initial population is initialized with a uniformly distributed random real number $R \in [0.0,\ 1.0]$.

Step 2. Cycles provided by each chromosome are ordered according to a nested (or inverted-nested) policy by using simple *QuickSort* sorting algorithm. For example, if an initial phenotype is defined as <2, 8, 4, 1>, then a phenotype with nested policy constraints is expressed by <8, 4, 2, 1>.

Step 3. Order-up-to levels S_i are calculated for each phenotype using the following sequence of analytical formulas:

$$\mu_{DDLCy_{k,i}} = \mu_{d_{k,i}} * (Cy_i + \mu_{L\, j\in i}) \qquad (4)$$

$$\sigma_{DDLCy_{k,i}} = \sigma_{d_{k,i}} * \sqrt{(Cy_i + \mu_{L\, j\in i})} \qquad (5)$$

$$SS_i = NORMSINV(CSL_i) * \sqrt{\sum_{k\in succ_i} \sigma^2_{DDLCy_{k,i}}} \qquad (6)$$

$$S_i = \sum_{k\in succ_i} \mu_{DDLCy_{k,i}} * SS_i \qquad (7)$$

where $\mu_{dk,i}$ is average demand received at stock point i for stock point k; $\sigma_{dk,i}$ is standard deviation of demand received in stock point i for stock point k; μ_{Lj} average lead time of process j, $\mu_{DDLCyk,i}$ average demand of end-customer k to stock point i during lead time and replenishment cycle; $\sigma_{DDLCyk,i}$ standard deviation of demand of end-customer k to stock point i during lead time and replenishment cycle; SS_i safety stock of stock point i; *NORMSINV* is Excel statistical function; CSL_i is customer service level of stock point i.

Step 4. Average total costs and average fill rate are calculated for each chromosome in the population through simulation of a supply chain. Example of linear network simulation model is given below. Number of simulation replications and a length of warm-up period are defined using a Welch-method.

Step 5. To assign fitness values to chromosomes, the above described Pareto-ranking NSGA-II algorithm is used.

Step 6. The resulting population is sorted in increasing order according to fitness values.

5.4. Population sizing

The population size should guarantee both adequate genetic diversity and reasonable simulation processing time. It is chosen in such a way that every solution in the search space is attainable with the crossover genetic operator [15]. The following formula for binary-encoded chromosomes [12] is used in the proposed algorithm:

$$\text{Population size} = 1.65 * 2^{(0.2*length)}, \qquad (8)$$

where *length* is a length of bit string.

5.5. Crossover and mutation operators

One of the major issues of genetic operators is the relative importance of two genetic operators: *crossover* and *mutation* since they provide the exploration and exploitation of a search space. The exploration is used to investigate new and unknown areas in a search space. The exploitation is aimed to making use of knowledge acquired by exploration to reach better positions on the search space. Pure random search is good at exploration, but has no exploitation. Hill climbing is good at exploitation but has little exploration [16]. An effective trade-off between exploration/exploitation is a key issue in design of GA. The population of GA should have some "excellent" individuals to guide the search directions, i.e. exploration. On the other hand, it should have proper population diversity to avoid the premature convergence, i.e. exploitation. Different adaptive techniques have been introduced to provide a balance between exploration and exploitation and attempt to solve the problem of finding optimal parameters [17].

Crossover design analysis. The crossover operator must have the ability to propagate so called building blocks throughout the population and to create new ones [16].

For example, Holland [18] provided the initial formal analysis of the behaviour of GAs and investigated *1-point* crossover operator. Spears and De Jong [19] extended Holland's analysis to *multi-point* and *uniform* crossover. De Jong hypothesized that a multi-point crossover could produce the schema disruption and result in a pure random search. Uniform crossover developed by Syswerda is one of the most promising and widely used crossover operators, because it (i) inherits common genes and (iii) results in a minimally biased exploration of the space being searched [20], [21]. Warwick [16] proposed the uniform adaptive crossover, which is based on on-line and off-line performance measures. Vekaria and Clack [17] proposed a selective crossover, which overcomes two-point and uniform crossovers on a set of test problems that contain characteristics common in practical problems.

Currently, the uniform crossover is introduced in MOSGA. It operates in the following way:

1. Select randomly one of two parents;
2. Select a current gene of a child and make it equal to the gene of selected parent;
3. Continue until all genes are defined.

Mutation design analysis. Mutation is used by GAs to diversify chromosomes in order to exploit the search space as widely as possible. Since the binary encoding is used in MOSGA, the mutation consists just in flipping random bits of the genotype.

Probability of crossover and mutation. There are three classes of methods of controlling rates of genetic operators, namely, deterministic, adaptive and self-adaptive parameters control. The first class changes the rate with time according to some deterministic rule with no feedback from the search. The second class on the other hand uses feedback from the search to modify the rate. In the third class the rate is evolved itself along with the search variables. Adaptive parameter control is preferred due to its adaptability to the search and its controllability [22]. For example, Song et al. [24] proposed a fuzzy logic controller to adapt the crossover rate. Van Hemert and Bäck [25] introduced the re-sampling ratio, which shows how much a search algorithm is re-visiting points in the search space, and it is defined as the number of points revisited divided by the total number points visited. The adaptive crossover control is planned to be introduced in MOSGA.

The mutation probabilities are expected to be low at the beginning of the search, since high values may cause strong disruption of promising schemata and therefore steer the algorithm away from the most promising regions of the search landscape [26].

In this paper, dynamic adjustment of the mutation rate introduced in *Pikaia* software is implemented [27]. It is based on a measure of the degree of convergence of the population:

$$\Delta S = \frac{S(r=1) - S(r = npop/2)}{S(r=1) + S(r = npop/2)}, \qquad (9)$$

where $S(r{=}1)$ and $S(r{=}npop/2)$ are a fitness of the best and median chromosomes. If ΔS is smaller (larger) than a predetermined level, then the mutation rate is increased (lowered).

5.6. Selection strategy

Selection strategy decides which areas of a search space to focus efforts on. Selection depends on:

- Comparison operator;
- Type of selection.

The selection procedure of MOSGA is taken from NSGA-II algorithm. As a result, every chromosome i has two attributes, such as:

- Non-domination rank, r_i;
- Crowding distance, d_i.

A crowded comparison operator $\geq_n$ is defined as follows [3]:

$$i \geq_n j \ \text{if} (r_i < r_j) \,\text{or}\, ((r_i = r_j) \,\text{and}\, (d_i > d_j)) \qquad (10)$$

Thus, between two solutions with differing non-domination ranks, the point with the lower rank is preferred. If both the points belong to the same frontier, then the point, which is located in a region with smaller number of points, is preferred [14].

The type of selection is very important for efficient investigation of a search space. It is known that proportionate selections are dependent on the objective function used. In this paper a tournament selection is used, because, as Falkenauer [28] has concluded, it applies significant selection pressure while avoiding the pitfalls of fitness ordering or ranking. Moreover, the tournament selection has the smallest time complexity. The idea is simple enough:

1. Choose randomly chromosomes from a population. Number of chromosomes is defined by a tournament size;
2. Select the best chromosome from this group for further genetic processing using comparison operator (10);
3. Repeat until the mating pool is filled.

Tournaments are held between pairs of chromosomes (tournament size = 2).

5.7. Reproduction strategy

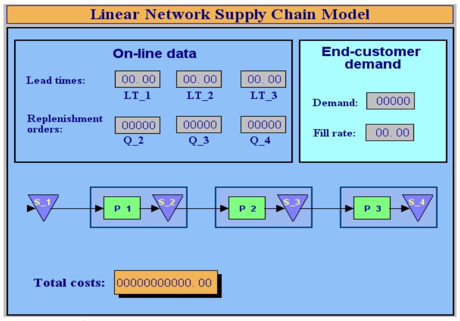
A reproduction strategy is used to control ways, in which new chromosomes are to be incorporated in the population. MOSGA applies a *full generational replacement*, which means that once all offspring have been produced, the entire parent population is replaced by the offspring population. After that a new iteration begins. MOSGA utilizes the elitism, which is the process of evolving chromosomes with a bias towards the better ones. Elitism is important since it allows the solutions to get better over time.

5.8. Termination criterion

In the MOSGA framework, the algorithm tracks the population diversity and stops the optimisation, when it falls below a preset threshold. Here, the meaning of diversity relates to objective functions values. If there is no change in the best values for some number of generations, then the algorithm is terminated.

6. Illustrative example

The above example illustrates a prototype of the MOSGA algorithm applied to the linear network supply chain cyclic planning. Main screenshots present the network simulation model (Figure 2), MOSGA user interface (Figure 3) and summary of results (Figure 4 and 5).



Linear Network Supply Chain Model

On-line data

Lead times: 00. 00 LT_1, 00. 00 LT_2, 00. 00 LT_3

Replenishment orders: 00000 Q_2, 00000 Q_3, 00000 Q_4

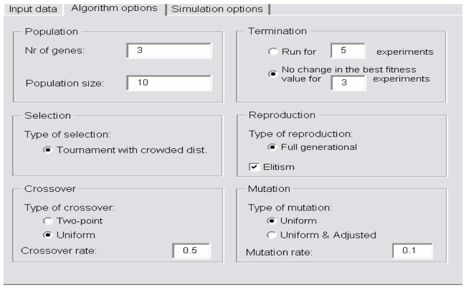
End-customer demand

Demand: 00000

Fill rate: 00. 00

Total costs: 00000000000. 00

Figure 2. Linear network simulation model



Population

Nr of genes: 3

Population size: 10

Termination

○ Run for 5 experiments

● No change in the best fitness value for 3 experiments

Selection

Type of selection:

● Tournament with crowded dist.

Reproduction

Type of reproduction:

● Full generational

☑ Elitism

Crossover

Type of crossover:

○ Two-point

● Uniform

Crossover rate: 0.5

Mutation

Type of mutation:

● Uniform

○ Uniform & Adjusted

Mutation rate: 0.1

Figure 3. MOSGA user interface

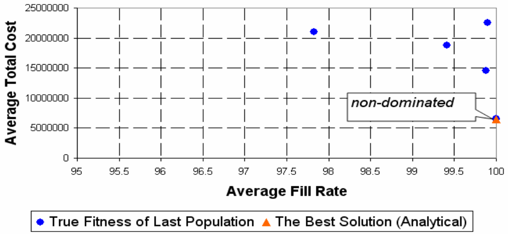


Figure 4. Comparison between exhaustive enumeration and MOSGA solutions

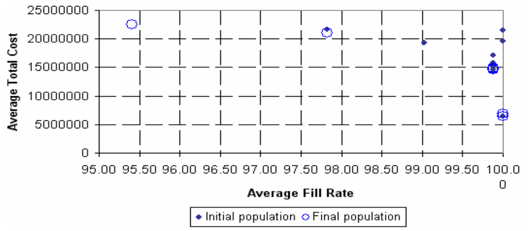


Figure 5. Comparison between initial and final population solutions mapped in the objective space

Because of the example's simplicity motivated by the necessity to verify the algorithm, the final Pareto front degraded to a single solution. Nevertheless, optimisation results are similar to exhaustive enumeration that proves the algorithm's efficiency.

7. Conclusions

The paper presents the theoretical framework for the development of multi-objective simulation-based optimisation genetic algorithm for supply chain cyclic planning and optimisation. To design this algorithm, the features of fitness assignment procedure, encoding mechanism, crossover operator, mutation operator, and selection and reproduction strategies are discussed in the paper. Finally, a simple numerical example that illustrates the MOSGA prototype is given.

Future research will focus on the improvement of MOSGA algorithm and its application to complex supply chain network. It will include screening experiments for excluding insignificant decision variables and reducing a search space. Besides that, the sensitivity analysis of the proposed algorithm is planned.

8. Acknowledgments

The presented research is funded by the ECLIPS Specific Targeted Research Project of the European Commission "Extended Collaborative Integrated Life Cycle Supply Chain Planning System".

The authors would like to thank Jonas Hatem from Möbius Ltd. for helpful comments and discussions.

9. References

[1] Merkuryeva, G. and. Napalkova, L. *Development of simulation-based environment for multi-echelon cyclic planning and optimization*. In: 6th EUROSIM Congress on Modelling and Simulation. Ljubljana, Slovenia, 2007, paper ID 452.

[2] G. M. Campbell and V. A. Mabert. *Cyclical Schedules for Capacitated Lot Sizing with Dynamic Demands*. In: Management Science, 1991, pp. 409 – 427.

[3] Y. Merkuryev, G. Merkuryeva, B. Desmet, and E. *Jacquet-Lagrèze. Integrating Analytical and Simulation Techniques in Multi-Echelon Cyclic Planning*. In: Proceedings of the First Asia International Conference on Modelling and Simulation (AMS 2007), Phuket, Thailand, 2007.

[4] Chen, J.-H. *Theoretical Analysis of Multi-Objective Genetic Algorithms – Convergence Time, Population Sizing and Disequilibrium*. In: Report for IEEE NNS Walter Karplus Research Grant, 2003.

[5] Dimopoulus, C. *A Novel Approach for Solution of the Multiobjective Cell-Formation Problem*. In: 18th International Conference on Production Research, Salerno, Italy, 2005.

[6] Jaszkiewicz, A. *Multiple objective metaheuristic algorithms for combinatorial optimization*. Habilitation thesis, Poznan University of Technology, Poznan, 2001.

[7] Veldhuizen, V. A. *Multi-objective Evolutionary Algorithms: Classifications, Analysis and New Innovations*. Dissertation, 1999.

[8] Farina, M. *Cost Effective Evolutionary Strategies for Pareto Optimal Front Approximation in Multiobjective Shape Design Optimisation of Electromagnetic Devices*. Doctorate Dissertation, 2001.

[9] Azadivar, F. *A Tutorial on Simulation Optimization*. In: Proceedings of the 1992 Winter Simulation Conference, 1992, pp. 198-204.

[10] Whitley, D. A Genetic Algorithm Tutorial.

[11] Acan, A. and Ünveren, A. *Evolutionary Multiobjective Optimization with a segment-Based External Memory Support for the Multiobjective Quadratic Assignment Problem*. In: Congress on Evolutionary Computation, 2005, pp. 2723- 2729.

[12] D. E. Goldberg. *Optimal initial population size for binary-coded genetic algorithms*. Technical Report TCGA-850001, University of Alabama, 1985.

[13] Srinivas N., Deb K., *Multi-objective optimization using nondominated sorting in genetic algorithms*. Evolutionary Computation, 1994, pp. 221-248.

[14] Deb, K., Agrawal, S. and Meyarivan, T. *A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II*. In: Proceedings of the Parallel Problem Solving from Nature VI Conference, 2000.

[15] Hatem, J. *On the use of evolutionary algorithms for the scheduling of call centre operators*. In: Proceedings of Simulation in Industry and Services, 2006.

[16] Warwick, T. J. *A GA Approach to Constraint Satisfaction*. Doctoral thesis, 1995.

[17] Vekaria, K. and Clack, C. *Selective Crossover in Genetic Algorithm: An Empirical Study*. In: Proceedings of the 5th International Conference on Parallel Problem Solving from Nature, 1998, pp. 438-447.

[18] Holland, John H. *Adaptation in Natural and Artificial Systems,* The University of Michigan Press, 1975.

[19] Spears, W. M. and De Jong, K. A. *An Analysis of Multi-point Crossover*.

[20] Senaratna, N. *Genetic Algorithms: The Crossover-Mutation Debate*, 2005.

[21] Kenneth A. De Jong and William M. Spears. *On the Virtues of Parameterized Uniform Crossover*. In: Proceedings of the Fourth International Conference on Genetic Algorithms, 1991.

[22] Sewell, M., Samarabandu, J., Rodrigo, R. and McIsaac, K. *The Rank-scaled Mutation Rate for Genetic Algorithms*, In: International Journal of Information Technology, 2006, pp. 32-36.

[24] Song, Y. H., Wang, G. S., Johns, A. T. and Wang, P. Y. *Improved genetic algorithms with fuzzy logic controlled crossover and mutation*. In: International Conference on Control, 1996, pp. 140-144.

[25] Van Hemert, J. I., Bäck, T. *Robust Parameter Settings for Variation Operators by Measuring the Resampling Ratio: A Study on Binary Constraint Satisfaction Problems*. Journal of Heuristics 10, 2004, pp. 629-640.

[26] Alvarez, G. *Can we make genetic algorithms work in high-dimensionality problems?*, 2002.

[27] Charbonneau, P. and Knapp, B. *A User Guide to Pikaia*. 1995.

[28] Falkenauer, E. *Genetic algorithms and grouping problems*, 1998.