ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6. Available from http://www.i6doc.com/en/.

Order Crossover for the Inventory Routing Problem

Mohamed Salim Amri Sakhri[1], Mounira Tlili[2], Hamid Allaoui[3], Ouajdi Korbaa[4]

1- Supérieur de Gestion de Tunis - Laboratoire Strategies for Modelling and ARtificial inTelligence - Le Bardo 2000 - Tunisie

2- Institut Supérieur de Transport et de la Logistique de Sousse - Sousse 4023 - Tunisie

3- Université d'Artois - Laboratoire de Génie Informatique et d'Automatique de l'Artois - Béthune 62400 - France

4- Institut Supérieur Informatique et Techniques de Communication de Sousse - Laboratoire Modeling of Automated Reasoning Systems - Hammam Sousse 4011 – Tunisie

Abstract. In this paper, we aim to find a solution that reduces the logistical activity costs by using new hybrid meta-heuristics. We develop, in this work, a genetic algorithm (GA) with a hybrid crossing operator. The operator considered is the Order Crossover (OX); we will test our hybrid algorithm in a Periodic Inventory Routing Problem (PIRP). Our study proves the performance of the hybrid operator OX compared with the classic GA, demonstrate the competitiveness of this innovative approach to solve the large-scale instances and bring a better quality of the solution.

1 Introduction

One of the logistical problems that have seen a remarkable development, nowadays, is the Inventory Routing Problem (IRP) introduced, for the first time, by [1]. IRP counts among the NP-hard Problem. IRP become a very common problem studied by the researchers because of his benefits in the economic life. We quote [2] and [3], who's studied different structures of IRP. These authors have used GA to solve their problem, showing the efficiency of this meta-heuristic in solving the combinatorial problem. In our case, we have chosen the use of GA with a hybrid crossover operator called Order Crossover (OX) instead of the classic two-point crossover operator. This technique has shown its effectiveness in some cases of VRP studies [4] and [5]. This operator was not cited in the searches that touches the IRP, this has aroused our interest to study the fallout of this method on an IRP. The problem studied, in our paper, is to avoid stock-outs at the customers while optimizing total costs in the supply chain. Our IRP can handle the following issues: inventory management of each customer and at the supplier, delivery quantities to avoid stock-outs, customer allocation to delivery periods, design and route optimization. The benchmarks considered, in this study, are the classic instances of [6] and the new instances proposed in [7], where they consider deterministic demands and only one vehicle available at the supplier.

In the following, we will start by describing formally our problem; we present a mathematical programming formulation for the PIRP. The third section is devoted to introducing, briefly, the hybrid genetic algorithm developed to solve our proposed problem. In the fourth section, we present the results of extensive computational experiments, we analyze the relations between inventory and transportation costs and

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6. Available from http://www.i6doc.com/en/.

we compare our solutions against the ones from the literature. The conclusion and perspectives are provided in the last section.

2 Optimization of IRP

In this paper, we are interested in studying, analyzing and solving a multi period IRP. Our goal is to optimize the total costs in the supply chain and avoid stock-outs for the customers. In the following, we begin by describing our studied IRP that will allow us, then to set up the mathematical model.

2.1 Presentation of Supply IRP

The problem can be represented by a graph consisting of a set of nodes $N = \{0..., n\}$, where the node 0 is the supplier, and the subset $N' = N \setminus \{0\}$ represents the customers. At each discrete time over a finite horizon $t \in T$, a quantity r_0^t will become available at the supplier and a quantity r_i^t is consumed by the customer $i \in N'$. Y_{ij}^t is a binary variable equal to 1 if the edge $(i, j) \in N$ is traversed by the vehicle at period $t \in T$, otherwise 0. C_{ij} is the cost of the route between customers i and j. The variable q_i^t is the quantity delivered to a customer i at period $t \in T$. The inventory holding costs are h_i units for each node $i \in N$ per period. The variables N_i^t is used in our model to indicate the inventory level at node $i \in N$ at each period $t \in T$. The supplier and the customers have a predefined initial inventory level equal respectively to N_0^0 and N_i^0. For each customer $i \in N'$ there is a maximum and minimum level of inventory denoted by U_i and L_i. The binary variable x_i^t is equal to 1 if node $i \in N$ is visited at period $t \in T$, otherwise 0. Q is the capacity of the vehicle, in our case, we use only one vehicle. The variable v_i^t is the quantities that the vehicle carries after delivering to the retailer i in each period.

2.2 Mathematical Modeling of Supply IRP

The mathematical model is inspired from [8] and formulated as follows:

$$Min \sum_{t=1}^{T} \sum_{i=1}^{N} h_i N_i^t + \sum_{t=1}^{T} \sum_{i=1}^{N} \sum_{j=1}^{N} C_{ij} Y_{ij}^t \quad (1)$$

$$N_0^t = N_0^{t-1} + r_0^t - \sum_{i=1}^{N'} q_i^t \quad (2)$$

$$N_i^t = N_i^{t-1} - r_i^t + q_i^t \quad (3)$$

$$L_i \leq N_i^t \leq U_i \quad (4)$$

$$q_i^t \geq U_i x_i^t - N_i^{t-1} \quad (5)$$

$$q_i^t \leq U_i - N_i^{t-1} \quad (6)$$

$$q_i^t \leq U_i x_i^t \quad (7)$$

$$\sum_{i=1}^{N'} q_i^t \leq Q x_0^t \quad (8)$$

$$\sum_{i=1}^{N} Y_{ij}^t = \sum_{i=1}^{N} Y_{ji}^t \quad (9)$$

$$v_i^t - v_j^t + Q Y_{ij}^t \leq Q - q_j^t \quad (10)$$

$$q_i^t \leq v_i^t \leq Q \quad (11)$$

$$(Y_{ij}^t, x_i^t) \in \{0, 1\} \quad (12)$$

$$Y_{0j}^t \in \{0, 1, 2\} \quad (13)$$

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6. Available from http://www.i6doc.com/en/.

$$(q^t_{i\in N'}, v^t_{i\in N'}) \geq 0 \quad (14)$$

The objective function (1) is to minimize the total operating cost that is the sum of the inventory costs at the supplier and at the customers, and the costs of the routes over the time horizon. The inventory level of the supplier and the customers at the period t are guaranteed by the constraints (2) and (3). Constraint (4) enforce the inventory level to stay between the lower bound and upper bound. Constraints (5) to (7) introduced by [8] ensure the Order-Up-To-level (OU) policy. The constraint (8) is the vehicle capacity constraint. The constraint (9) ensures the flow conservation. Sub-tour elimination constraints are (10) and (11) and constraints (12) to (14) are the non-negativity and integrality constraints.

3 Order Crossover Genetic Algorithm (OXGA)

In this section, we will describe the algorithm OXGA which use a hybrid crossover and classic mutation operators. We will use this algorithm to determine the best heuristic that can solve better our IRP. Our optimization process is developed as follows.

3.1 Construction of an initial solution

The main objective of the construction phase is to determine the customer set that enables economies and avoids stock-outs during each period. This step considers both the demand and the cost of storage at the customers. The construction of the initial solution is a way to control the quantity of stock, at the different nodes of the distribution network, to avoid stock-out in each period and minimize the total storage costs, this simplifies the answer to our desired plan of experiments.

Assignment of customers at each period: This step will help us to define the set of customers to be served every period. In the algorithm, the customers are ranked in the non-decreasing order of the average number of time units needed to consume the quantity $U_i - L_i$ and the customers with the same number of time units are ranked in the non-increasing order of storage cost. When retailer *i* is considered, a set of delivery time instants are determined. **The generation of the initial population:** This step is used to generate an initial population of delivery routes as a basis for future generations. The set of delivery routes that compose the initial solution is generated randomly. **Evaluation function:** The individuals of the evaluation function, in our work, are the delivery routes. The quality of an individual is reflected in the fitness function. This quality of the generated solution should be compared with the best initial solution. The travel distance and the inventory level are referred to as the fitness of the individual. The fitness relative to individuals must be determined as necessary for the selection and replacement steps. **The selection operator:** We choose, in our work, the uniform selection operator as the selection is made randomly, uniformly and without the intervention of the adaptive value.

3.2 Improving the delivery routes

This step aims to optimize only the distance traveled without consideration of other objectives defined in the previous step. We will start by performing the OX operator, then the classic mutation process, and finish with the selected technic of replacement.

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6. Available from http://www.i6doc.com/en/.

In the following, we will describe the different function process of the selected operators.

Order Crossover Operator (OX): A two-point crossover is done between 1 and N where N is the length of the chromosome and the genes are coded with integer numbers. We select randomly the positions of the two cut points. Fig. 1 shows the OX process to constructs the offspring. First, the genes are copied down between the cuts with similar way into the offspring. Then, starting from the second cut point of one parent, the genes from the other parent are copied in the same order omitting existing genes. The sequence of the genes in the first parent from the second cut point is "8-4-5-1-2-7-3-6". After removal of genes 7, 3, 1 and 5, which are already in the second offspring, the new sequence is "8-4-2-6". This sequence is placed in the first offspring starting from the second cut point. The whole procedure can be implemented for the second offspring.

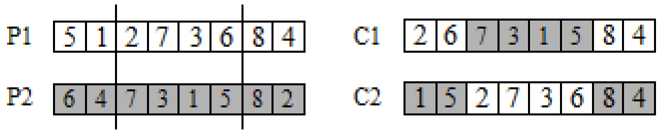

Fig. 1: Order CrossOver OX process

Mutation: The mutation is considered to be an operator responsible for maintaining the genetic diversity of the population. The performance of our GA has been improved by applying a simple mutation to the new generation, in which two genes are randomly selected and their values are exchanged. Thus, an exchange of position is realized between two randomly selected customers. **The replacement operator:** We used, in our work, the selection tournament technique to reintroduce the offspring. This technic uses random selection of parent pairs, and each pair of parent and his offspring fitness will be evaluated. Then, we choose the pair of individuals who has the evaluation highest score: we select the pair of chromosomes from the set of individuals (P1, P2, C1 and C2) that generate lower costs with a TRI function. **Stop condition:** The algorithm stops when a predefined number of generations is reached.

4 Experiments and Results

In this section, we test the performance of our algorithms implemented in Java EE-like for ECLIPSE environment and run on a PC with a 2.40 GHz Intel® Core™ i7-5500U CPU @ processor and 8 GB of RAM such as: GA represents the results obtained with Pc = 0.5, Pm = 0.1 which are, respectively, the probabilities of the classic two-point crossover and mutation; OXGA represents the results obtained with Pox = 0.5, Pm = 0.1 which are the probabilities of the order crossover and mutation. A compilation of each heuristic was generated. The initial population for each heuristic is equal to 50 pathways and the number of iterations is equal to 200, in the case of instances up to 50 customers. For the 30 large instances, up to 200 customers, the number of paths for the initial solution and the number of iterations, are respectively 100 and 500. For each heuristic, the best results for each set of instances were averaged following a test cases.

Table 1 shows the results obtained by our algorithms and those of [7] and [9] who have used the small-instances of the benchmark of [6]. The solutions in bold are the best solutions of the approximated algorithms HAIR, ALNS, GA and OXGA.

Instance	z*	HAIR	ALNS	GA	OXGA	BS
T = 3						
Small-5	1418,76	**1418,76**	**1418,76**	**1418,76**	**1418,76**	1434,09
Small-10	2228,67	2228,73	**2228,67**	**2228,67**	**2228,67**	2245,61
Small-15	2493,47	**2493,47**	**2493,47**	**2493,47**	**2493,47**	2555,22
Small-20	3053,02	**3053,56**	3055,58	3121,43	3121,43	3176,92
Small-25	3451,15	**3451,15**	3451,86	**3451,15**	**3451,15**	3552,08
Small-30	3643,22	3643,99	3645,70	**3643,38**	**3643,38**	3774,21
Small-35	3846,87	**3848,46**	3850,83	3958,73	3958,73	4022,05
Small-40	4125,70	**4128,51**	4140,16	4191,68	4150,79	4394,94
Small-45	4270,61	**4276,89**	4283,33	4283,33	4279,19	4594,91
Small-50	4810,87	**4831,97**	4841,26	4887,16	4887,16	5090,68
T = 6						
Small-5	3299,98	**3299,98**	**3299,98**	**3299,98**	**3299,98**	3348,43
Small-10	4832,89	**4832,89**	**4832,89**	**4832,89**	**4832,89**	4899,86
Small-15	5566,39	**5566,39**	5582,80	5638,59	5638,59	5803,08
Small-20	6833,29	**6838,42**	6857,90	6920,31	**6838,42**	7035,02
Small-25	7454,15	**7471,42**	7478,80	7475,88	7475,88	7913,47
Small-30	7847,39	7892,29	**7888,56**	7899,12	7899,12	8214,21

Table 1: Average solution values for the small-instances

The hybridization of our GA affects the quality of the obtained solution. The results are very competitive by comparing OXGA to GA, but OXGA produces less competitive results compared to HAIR of [7] with -17.55% and -16.60% in the case of ALNS of [9] when T = 3. The results of our algorithms are less good with lower solutions off -6.36% compared to HAIR and -3.45% compared to ALNS, but OXGA remains better than the classic GA because it improves the results of 7.14 % when T = 6. Indeed, in more than 30 cases out of 80, our algorithm provides optimal solutions. In Table 1, the effectiveness of evolutionary methods and, above all, GAs compared to local search methods in terms of distance reduction and storage cost are not visible in the case of small instances. But our modified algorithm is more efficient than conventional GA because it gives either the best results or the same results proposed by the GA. It is possible to go further and test our algorithm on larger instances.

We continue to omit OX Operator. Table 2 shows the average results obtained with HAIR, GA and OXGA, when the number of customers reaches 200 and T = 6.

Instance	HAIR	GA	OXGA
Large-50	10513,07	10527,67	**10494,89**
Large-100	15613,61	15931,58	**15291,50**
Large-200	24787,14	25306,66	**24340,32**

Table 2: Average solution values for the large-instances

According to Table 2, the total cost has been improved with OXGA, which has led to better results compared to the results of HAIR. Indeed, these results make it possible to conclude that OXGA and HAIR produce very similar results in some cases when the number of customers is equal to 50 or 100, and better results reaching a reduction of 1.80%, in the case of the number of clients is equal to 200. The use of OX allows us to save on average 1.54% of the total cost. These experiments highlight the fact that if we increase the number of generation and/or execution, this will provide reliable results and converge towards the optimal solution of the problem. Table 2 also gives the average results obtained with the conventional GA and OXGA with the same calculation conditions. By comparing their results, we note that OXGA provides for most of instances the best results. These results prove that the hybridization of the crossover operator is beneficial and can solve optimally the instances up to 200 clients.

5 Conclusion

In this paper, we are studied a typical vehicle routing problem that presents one of the most sought-after multi-objective problems by logistics researchers. It encompasses two major areas of research that are vehicle routing and stock management, in a distribution network. To solve the problem, we have implemented a hybrid genetic algorithm OXGA. Our algorithm has shown its efficiency to solve large instances by providing the best results on the tested instances so far. This research will be an opening of research and investigation on the development plan of the new hybrid algorithms either by changing the crossover or mutation operators by other operators to test the effectiveness of such an operation and testing other distribution network structures.

References

[1] A. Federgruen and P. H. Zipkin, A combined vehicle routing and inventory allocation problem. Operations Research, 32(5):1019–1037, 1984.

[2] A. Nevin, A Genetic Algorithm on Inventory Routing Problem. Artvin Çoruh University, Emerging Markets Journal, Volume 3 No 3, ISSN 2158-8708 (online), DOI 10.5195/emaj.2014.31, 2014.

[3] Y.B. Park, A genetic algorithm for the vendor-managed inventory routing problem with lost sales. Expert Systems with Applications, 2016.

[4] C. Prins, A simple and effective evolutionary algorithm for the vehicle routing problem. Computers & Operations Research, 31(12) :1985-2002, 2004. 25, 26, 28, 30, 32, 33, 2004.

[5] M. Haj-Rachid, W. Ramdane-Cherif, C. Bloch and C. Chatonnay, Comparing the performance of genetic operators for the vehicle routing problem. Conference on Management and Control of Production Logistics, At University of Coimbra, Portugal, Volume: 313-319, 2010.

[6] C. Archetti, L. Bertazzi, G. Laporte and M.G. Speranza, A branch-and-cut algorithm for a vendor-managed inventory-routing problem. Transportation Science, 41:382–391, 2007.

[7] C. Archetti, L. Bertazzi, G. Paletta and M.G. Speranza, Analysis of the maximum level policy in a production-distribution system. Computers & Operations Research, 38:1731–1746, 2011.

[8] L. Bertazzi, P. Giuseppe and S.M. Grazia, Deterministic Order-Up-To Level Policies in an Inventory Routing Problem. Transportation Science, 36(1):119-132, 2002.

[9] L.C. Coelho, J.F. Cordeau and G. Laporte, Consistency in multivehicle inventory routing. Transportation Research Part C: Emerging Technologies, 24:270–287, 2012.