

Chemotaxis Differential Evolution Optimization Algorithm (CDEOA) for Minimization of Supply Chain Cost with Embedded Risk

Yunus Emre Yıldız¹, Oğuz Altun²

¹Department of Computer Engineering, Epoka University, Tirana, Albania

²Department of Computer Engineering, Yildiz Tech. University, Istanbul, Turkey

Abstract: Minimization of supply chain cost is considered to be a real challenge due to its complexity. Exponential growth of a search space with a small rise in parameter values, huge number of parameters, and their distributions are the main factors that increase the difficulty of a supply chain. Chemotaxis Differential Evolution Optimization Algorithm (CDEOA) is an innovation and hybrid optimization algorithm which was tested on CEC 2014 benchmark functions for global optimization. In this paper CDEOA has been applied to minimize the supply chain cost with other well-known algorithms; Particle Swarm Optimization (PSO), Bacterial Foraging Optimization Algorithm (BFOA), Bat Algorithm (BA), Genetic Algorithm (GA), and Tabu Search (TS). The result of this study shows that the CDEOA outperforms, or is comparable to, its competitors in terms of accuracy and efficiency.

Keywords: Chemotaxis Differential Evolution Optimization Algorithm (CDEOA), Differential Evolution, Supply Chain Cost Problem, Nature inspired algorithms, Metaheuristics.

1. Introduction

A supply chain is a network of globally distributed business entities that performs the functions of procurement of materials, transformation of these materials into intermediate and finished products, and the distribution of these finished products to retailers or customers. Business entities that make up a supply chain are: suppliers, manufacturers, warehouses, retailers, and customers. Suppliers provide raw materials. Manufacturers transform the raw materials into final products. Warehouses distribute the products to retailers, which in turn sell the products to their customers [5]. The integration and coordination of entities are one major aspect of a supply chain because a decision in one element influences the entire chain.

Minimization of cost and maximization of profits for each business entities have unveiled a new optimization problem known as Supply Chain Cost Problem which minimizes the cost of a globally distributed supply chain and maximizes the profits of chain stakeholders [14]. When large numbers of decision variables and alternatives exist, these kinds of problems are identified as non-deterministic polynomial-time hard (NP-hard) problems and they need more complex optimization algorithms to guide the search for optimum or near-optimum solutions [9]. In this context, random search techniques have been popular in solving computationally complex (NP-hard) problems due to their ability to find effective solutions in a short amount of time. In this field, Castillo [2] proposed a novel capacitated Supply Chain Network Design (SCND) model which evaluates the overall economic profit of the supply chain with a metaheuristic-based approach. Castillo [13] also presented a comprehensive review by studying and analysing the application of metaheuristics to solve bioenergy supply chain models.

Chemotaxis Differential Evolution Optimization Algorithm (CDEOA) [11] is an innovative optimization algorithm which was inspired by *Escheria coli* bacteria movements. CDEOA is based on the combination of two new strategies into the chemotaxis step of BFOA: weak bacterium's search and strong bacterium's foraging. The

performance of weak bacteria is enhanced by randomly moving to new positions whereas the performance strong bacteria are enhanced by integrating the ideas of differential evolution Differential Evolution (DE) [8] operators.

The proposed algorithm was compared with Particle Swarm Optimization (PSO) [4], standard Bat Algorithm (BA) [3], Genetic Algorithm (GA) [10], Tabu Search (TS) [17], and Bacterial Foraging Optimization Algorithm (BFOA) [6] on five different scenarios of the Supply Chain Cost problem.

The remaining parts of this paper are organized as follows: Section 2 introduces the hybrid CDEOA. Section 3 outlines the supply chain cost problem. Section 4 includes a short summary of the simulation results and the performance comparison. Section 5 concludes the paper.

2. Chemotaxis Differential Evolution Optimization Algorithm (CDEOA)

The basic idea behind the proposed CDEOA relies on two different strategies: a) making “weak” bacteria more explorative, where “weak” bacteria are the ones in places with poor nutrient concentrations, and b) making “strong” bacteria more exploitative, where “strong” bacteria are the ones in places with rich nutrient concentrations [11].

If a bacterium finds a new, promising area and keeps running for a number M_r of generations, then this bacterium experiences exploitation state (line 59 in Algorithm 1). If a bacterium’s current fitness remains unaltered for a number M_t of generations, then this bacterium experiences exploration state (line 50 in Algorithm 1). The CDEOA performs a local search through the chemotaxis movement operation of BFOA and the global search over the search space through Random Search (RS) and DE operators.

2.1. Making Weak Bacteria Explorative

During the chemotaxis process of BFOA, a bacterium in the neighbourhood of the noxious substance will try to move to a place with rich nutrient concentration by taking larger steps [1]. In Yıldız’s [11] contribution, a bacterium performs an elimination process which leads it to change its position randomly after a number of unlucky movement attempts M_t . That is to say, if a bacterium’s number of unlucky attempts of taking tumble steps E_t approaches the maximum M_t , the bacterium is liquidated at random (line 50-51 in Algorithm 1). “Making weak bacteria explorative” process is performed by the elimination-dispersal process in the original BFOA. In CDEOA, the elimination-dispersal process of BFOA is replaced with “making weak bacteria explorative” strategy.

2.2. Making Strong Bacteria Exploitative

The bacterium runs for a period of time in the same direction as long as it finds better nutrient-rich concentrations and retains its position. To make the bacterium more exploitative, the bacterium is expected to exploit the gradient of the promising area in the neighbourhood of a nutrient-rich substance. Accordingly, each bacterium takes the mutation, crossover, and selection operators of DE (line 62-64 in Algorithm 1) only if the bacterium’s lucky number of run steps E_r approaches the maximum number of run steps M_r (line 61 in Algorithm 1).

Algorithm 1 Detailed pseudo-code of CDEOA. Comments start with “//”. The code we discuss in the text is in boldface.

```

1: Parameters:
2:    $p \leftarrow$  dimension of the search space
3:    $S \leftarrow$  total number of bacteria in the population
4:    $N_c \leftarrow$  number of chemotaxis steps
5:    $N_s \leftarrow$  swimming steps
6:    $N_{re} \leftarrow$  the number of reproduction steps
7:    $C(i) \leftarrow$  the run length unit
8:    $M_t \leftarrow$  maximum number of tumble steps
9:    $M_r \leftarrow$  maximum number of run steps
10:   $f \leftarrow$  objective function to be minimized
11: // Initialize some local variables
12:  $E_t \leftarrow 0$  // Bacterium’s unsuccessful tumble step
13:  $E_r \leftarrow 0$  // Bacterium’s successful run step
14:  $\theta_{best} \leftarrow$  random position in the search space
15:  $f_{best} \leftarrow f(\theta_{best})$ 
16:  $M_{fes} \leftarrow$  maximum number of function evaluations allowed
17:  $N_{fes} \leftarrow 0$  // current number of function evaluations
18: // Define a helper function  $J$  that will call the actual objective
    function  $f$ . This helper function also updates the  $N_{fes}$ ,  $\theta_{best}$ ,
    and  $f_{best}$  variables. This approach makes the rest of the
    algorithm cleaner. Depending on the programming language and
    programming paradigm that will be used, this helper function
    may be moved outside the CDEOA block, or may be a method
    of a class.
19: function  $J(\theta)$ :
20:    $v \leftarrow f(\theta)$ 
21:    $N_{fes} \leftarrow N_{fes} + 1$  // update number of FES
22:   if  $v < f_{best}$  then
23:      $\theta_{best} \leftarrow \theta$  // update global best position
24:      $f_{best} \leftarrow v$  // update global best function value

```

```

25:   Return  $\nu$ 
26: end // Function
27: while  $N_{fes} < M_{fes}$  do // FES control loop
28:   for  $k$  from 1 to  $N_{re}$  do // Reproduction loop
29:     for  $j$  from 1 to  $N_c$  do // Chemotaxis loop
30:       for  $i$  from 1 to  $S$  do // Tumble-Swim loop
31:          $J_{last} \leftarrow J(\theta(i, j, k))$  //  $J(\cdot)$  computes the fitness
32:          $\Delta(i) \leftarrow$  random vector within  $[-1, 1]$  // Tumble
33:          $\theta(i, j + 1, k) \leftarrow \theta(i, j, k) + C(i) * \frac{\Delta(i)}{\sqrt{\Delta^T(i) * \Delta(i)}}$  // Move
34:         if  $J(\theta(i, j + 1, k)) > J(\theta(i, j, k))$  then
35:            $E_t \leftarrow E_t + 1$ 
36:         // Swim:
37:         for  $m$  from 1 to  $N_s$  do // Swim loop
38:           if  $J(\theta(i, j + 1, k)) < J_{last}$  then
39:              $J_{last} = J(\theta(i, j + 1, k))$ 
40:              $\theta(i, j + 1, k) = \theta(i, j, k) + C(i) * \frac{\Delta(i)}{\sqrt{\Delta^T(i) * \Delta(i)}}$ 
41:            $E_r \leftarrow E_r + 1$ 
42:         else
43:            $m = N_s$  // Break from swim loop
44:         end // If
45:       end // Swim loop
46:     end // Tumble-Swim loop
47:   // Exploration loop
48:   for  $i$  from 1 to  $S$  do // Exploration loop
49:     // Take an exploration step for bacterium  $i$ 
50:     if  $E_t = M_t$  then
51:        $\theta(i, j + 1, k) \leftarrow$  random position
52:        $J_{last} = J(\theta(i, j + 1, k))$ 
53:       if  $J_{last} < J(i, j, k)$  then
54:          $J(i, j + 1, k) \leftarrow J_{last}$ 
55:       end // If
56:        $E_t = 0$ 
57:     end // If
58:   end // Exploration loop
59: // Exploitation loop
60: for  $i$  from 1 to  $S$  do // Exploitation loop
61:   if  $E_r = M_r$  then let bacterium undergo:
62:     DE mutation
63:     DE crossover
64:     DE selection
65:   end // If
66: end // Exploitation loop
67: end // Chemotaxis loop
68: //Reproduction
69:  $J_{health}^i = \sum_{j=1}^{N_c+1} J(i, j, k)$  // Compute the health of each bacterium
70: Sort bacteria cost  $J_{health}$  in ascending order. Let bacteria with the highest  $J_{health}$  values die and the remaining bacteria with the best values reproduce.
71: end // Reproduction loop
72: end // FES control loop
73: Return  $\theta_{best}$ 

```

3. Supply Chain Cost Problem

Supply chain is a system which combines the network of interconnected business processes in order to: (1) get raw materials; (2) convert these raw materials into finished products; (3) add value to these products; (4) deliver these products to retailers or customers; (5) facilitate information exchange among various business entities (e.g. suppliers, plants, warehouses, markets, and retailers). Its primary target is to boost operational efficiency, profitability, and the competitive position of a firm and its supply chain partners [16].

The main entities of a supply chain as shown in Fig. 1 are: suppliers, manufacturers, warehouses, retailers and customers. Suppliers provide the raw materials to the manufactures which in turn convert the raw materials to final products. The chain continues then with warehouses; warehouses deliver the products from manufacturers to retailers, and retailers sell these products to the final customers [7].

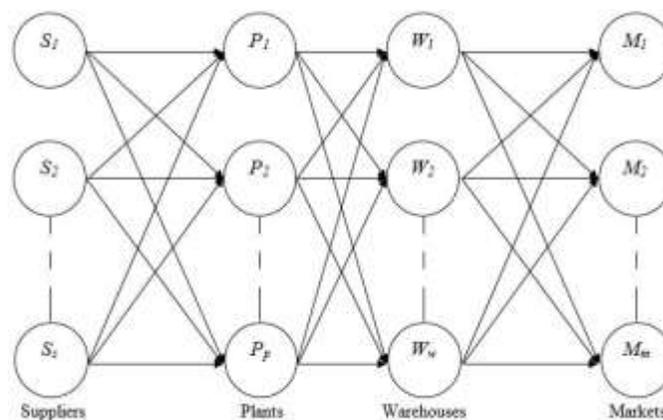


Fig. 1: Architecture of a supply chain.

Minimizing the total cost, maximizing the profit and fulfilling customers' needs while ensuring satisfaction has been studied by researchers [12] in terms of designing, analysing, and managing of supply chain. Many companies are concerned about analysing their supply chain as a whole system to improve their business. However, the process of analysing and managing the supply chain has been performed based on experience and intuition. This implies that finding the best supply chain strategies for a particular firm is a significant issue for industry. In this context, metaheuristics may play an important role in helping managers and consultants in the decision-making process [15].

The total cost of equation (1) of a globally distributed supply chain [12] is composed of supply cost of raw material (SCRM), cost of production (PC), cost associated with warehouses (WAC), and cost of markets (MC).

$$\text{Total Cost (TC)} = \text{SCRM} + \text{PC} + \text{WAC} + \text{MC} \tag{1}$$

The mathematical programming formulation that minimizes the total supply chain total cost (TC) is presented in equation (1) and considers all supplier, plant, warehouse, and market costs. Recently, metaheuristic algorithms have been broadly employed for optimizing NP-hard since they are simple, easy to implement, robust, and have been proven highly effective to solve complex problems [15].

4. Experimental Study

The CDEOA was tested with five different supply chain scenarios. In each scenario, production capacity of suppliers and plants, capacity of warehouses, and demand of each market were created randomly and were increased their complexity. The scenarios are presented in Table 1. In scenario 1: there are 2 suppliers, (s1 and s2), whose production capacities are 1000 and 1000 units; 3 plants, (p1 ,p2, p3) whose production capacities are 600, 400 and 400 units; 4 warehouses, (w1, w2, w3, w4) whose storage capacities are 400 300, 500 and 200 units; and 5 markets (m1, m2, m3, m4, m5) whose demands are 100, 100, 200, 70, and 30 units respectively.

TABLE I: Supply chain problem scenarios

	1	2	3	4	5
Suppliers	1000;1000	500;500;1000	500;500;1000	500;500 750;250	500;500;250 250;250;250
Plants	600;400;400	600;400;400	600;200 200;400	300;300;200 200;100;300	300;300;200;200 100;150;150
Warehouses	400;300;500;200	400;300;250 250;200	200;200;300 250;250;200	200;200;300 250;250;200	200;200;300 250;250;200
Markets	100;100;200 70;30	100;100;200 70;30	100;50;50 200;70;30	100;50;50 200;70;30	100;50;50 200;70;30

4.1. Compared Algorithms and Parametric Setup

The performance of the CDEOA was compared with BFOA, PSO, TS, BA, and GA. The study introduced in this paper aims to test the quality of final solution and convergence speed at the end of a fixed number of function evaluations (FES). The objective is to minimise the total cost of the supply chain operation which includes supplier cost, production cost, warehouse associated cost and market cost. Each scenario has a distinct number of dimensions depending on the complexity of the supply chain. In this context, scenarios have 38, 54, 72, 96, and 120 dimensions, respectively. The maximum number of FES was set to 1000. The population size of each algorithm was set to 20. Each algorithm was run 30 times.

For PSO, we employed the standard PSO and set the inertia weight $w = 1$ and acceleration coefficients $c_1 = c_2 = 2$ according to [4]. As for the BA, the algorithmic constants $\alpha = \gamma = 0.5$, frequency is in the range $[0, 2]$, and loudness $A_0 = 0.5$. For GA, mutation probability = 0.05 and crossover probability = 0.95. BFOA and CDEOA employ the same parameters as follows: $N_c = 100$, $N_s = 16$, $N_{re} = 8$, $C(i) = 0.1$.

TABLE II: Comparison of PSO, BFOA, TS, GA, BA, and CDEOA

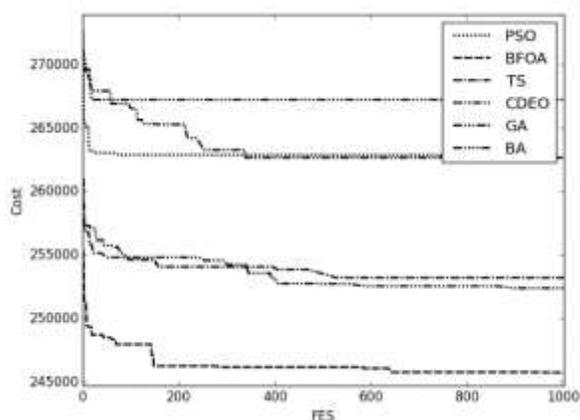
Scenarios	Dimensions	Algorithms	BCV	WCV	Mean	Stdev	Median	Time
1	38D	PSO	171774.07	406389.87	271898.87	61422.80	262674.67	2.75
		BFOA	158073.95	328604.65	256788.83	45965.09	245733.44	2.64
		TS	167997.11	448938.90	265207.46	60890.62	253209.65	3.26
		BA	149057.04	374563.73	264678.66	60600.65	267249.05	10.16
		GA	138873.38	319263.21	247192.37	46310.93	252379.22	3.30

		CDEOA	131436.66	483223.62	263664.37	66805.99	262662.79	3.65
2	54D	PSO	221549.15	424156.34	316044.20	50554.57	319207.67	16.89
		BFOA	177070.27	562895.49	339305.96	91896.03	323970.03	16.55
		TS	150616.39	468185.32	318433.36	73805.63	313026.23	21.98
		BA	173366.76	486141.78	319110.49	78046.28	308508.69	21.43
		GA	209731.19	498573.87	342671.21	65858.21	340691.22	22.19
		CDEOA	178180.85	536455.05	301219.47	79367.57	303121.86	16.79
3	72D	PSO	314819.83	698583.76	501751.33	101797.08	504006.41	5.59
		BFOA	301504.10	715808.25	500714.51	108195.15	508918.69	2.45
		TS	325904.98	657693.39	498861.93	85473.23	510978.49	3.05
		BA	289262.97	743135.80	488318.85	91721.61	490993.25	16.36
		GA	323051.57	725538.68	536753.61	82662.90	528767.20	6.24
		CDEOA	281094.41	739537.04	493928.04	115114.72	483599.93	3.07
4	96D	PSO	685639.86	870112.42	789663.58	63665.56	809801.15	4.83
		BFOA	530568.81	944208.98	741662.90	136583.12	742537.79	5.00
		TS	497928.05	981825.53	740919.58	149498.71	733425.54	5.55
		BA	491848.56	978094.90	710897.41	123887.43	683134.40	26.38
		GA	578853.52	1061163.49	760919.39	140865.37	700789.34	6.24
		CDEOA	477030.32	1019036.83	709229.15	159874.33	669061.34	5.01
5	120D	PSO	1007800.99	1619948.08	781593.63	164448.82	414210.04	28.54
		BFOA	1023073.99	1642535.82	777055.16	165842.10	392991.58	29.42
		TS	1038346.98	1665123.56	772516.68	167235.38	371773.11	30.29
		BA	1053619.97	1687711.30	767978.21	168628.66	350554.65	31.16
		GA	1068892.96	1710299.05	763439.74	170021.94	329336.19	32.04
		CDEOA	1084165.96	1732886.79	758901.27	171415.22	308117.73	32.91

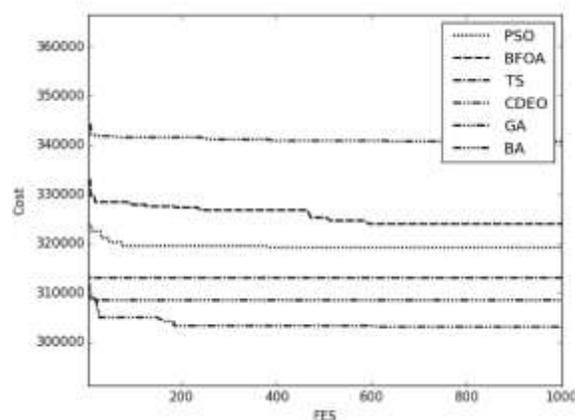
4.2. Experimental Study and Discussions

Table 2 reports the best cost value (BCV), the worst cost value (WCV), mean of the final best function values, the standard deviation of the final best function values (STDEV), median of the final best function values, and the mean time spent per trial in seconds.

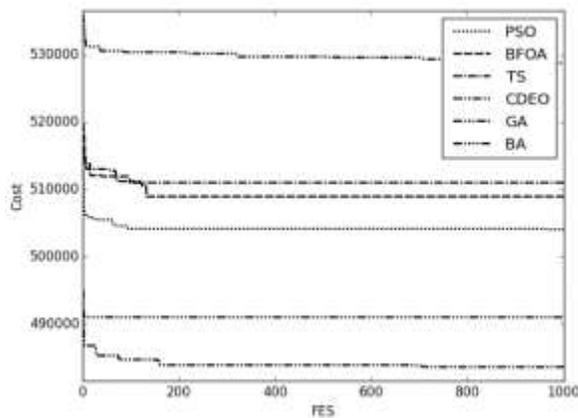
In Table 2, we can observe that CDEOA is superior overall to five algorithms in five different scenarios. We can infer the success of CDEOA in these scenarios may be due to its capability of balancing the exploration and exploitation with the aforementioned two CDEOA strategies. CDEOA outperforms its five competitors except GA in the first scenario. Worst cost value (WCV) of PSO shows better performance than CDEOA in all scenarios.



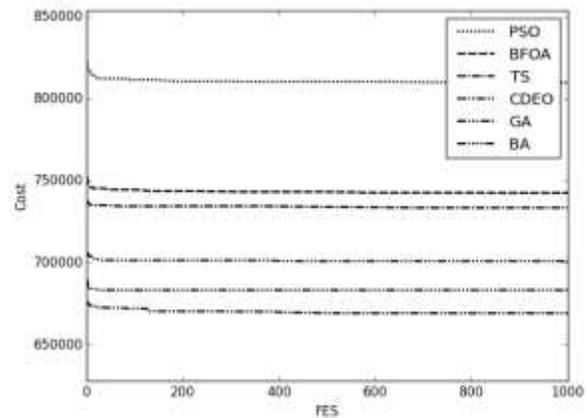
(a) Scenario 1



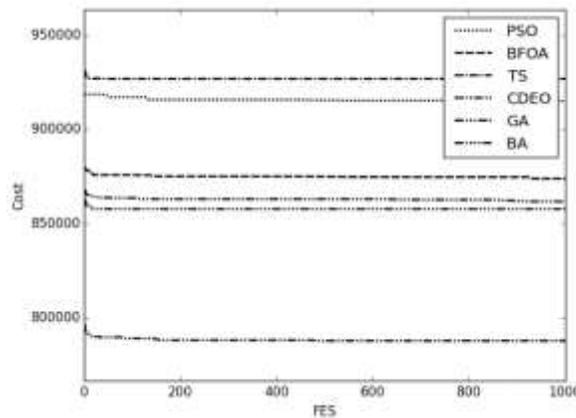
(b) Scenario 2



(c) Scenario 3



(d) Scenario 4



(e) Scenario 5

Fig. 2: Median convergence graphs of PSO, BFOA, TS, GA, BA, and CDEOA

In Fig. 2, the convergence map of PSO, BFOA, TS, GA, BA, and CDEOA shows that the CDEOA overall led faster convergence than its competitors in all scenarios. It did fail against BFOA, GA, and TS in the first scenario; however, these three algorithms were unable to maintain the same performance in the rest of the scenarios. In addition, we can also observe from scenario 1 of Table 2, the aforementioned algorithms slightly outperformed CDEOA.

5. Conclusion

CDEOA has been tested on a real life problem, supply chain. CDEOA is based on two strategies: making weak bacterium more explorative and making strong bacterium more exploitative, where two of which can be employed in any BFOA counterpart for performance improvement. CDEOA outperforms its competitors: PSO, BFOA, TS, GA, and BA in solving Supply Chain Cost problem.

The Python source codes of the algorithms PSO, BFOA, TS, GA, BA, and CDEOA is available in Oğuz Altun's repository (<https://bitbucket.org/oaltun/opn>).

6. References

- [1] S. Dasgupta, S. Das, A. Abraham, and A. Biswas, "Adaptive Computational Chemotaxis in Bacterial Foraging Optimization: An Analysis," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 4, pp. 919–941, Aug. 2009.

- <http://dx.doi.org/10.1109/TEVC.2009.2021982>
- [2] K. K. Castillo-Villar and J. F. Herbert-Acero, "A Metaheuristic-Based Approach for the Capacitated Supply Chain Network Design Problem Including Imperfect Quality and Rework," *IEEE Computational Intelligence Magazine*, vol. 9, no. 4, pp. 31–45, Nov. 2014.
<http://dx.doi.org/10.1109/MCI.2014.2350934>
- [3] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature inspired cooperative strategies for optimization (NICSO 2010)*, Springer, 2010, pp. 65–74.
http://dx.doi.org/10.1007/978-3-642-12538-6_6
- [4] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proceedings of the sixth international symposium on micro machine and human science*, 1995, vol. 1, pp. 39–43.
<http://dx.doi.org/10.1109/MHS.1995.494215>
- [5] R. Ganeshan and T. P. Harrison, "An introduction to supply chain management," *Penn State University, The United States*, 1995.
- [6] K. M. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," *Control Systems, IEEE*, vol. 22, no. 3, pp. 52–67, 2002.
<http://dx.doi.org/10.1109/MCS.2002.1004010>
- [7] J. T. Mentzer, W. DeWitt, J. S. Keebler, S. Min, N. W. Nix, C. D. Smith, and Z. G. Zacharia, "Defining supply chain management," *Journal of Business logistics*, vol. 22, no. 2, pp. 1–25, 2001.
- [8] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of global optimization*, vol. 11, no. 4, pp. 341–359, 1997.
<http://dx.doi.org/10.1023/A:1008202821328>
- [9] P. M. Pardalos, *Handbook of Combinatorial Optimization*. Dordrecht: Springer Verlag, 2005.
- [10] L. Davis and others, *Handbook of genetic algorithms*, vol. 115. Van Nostrand Reinhold New York, 1991.
- [11] Y. E. Yildiz and O. Altun, "Hybrid achievement oriented computational chemotaxis in bacterial foraging optimization: a comparative study on numerical benchmark," *Soft Computing*, pp. 1–17, 2015.
<http://dx.doi.org/10.1007/s00500-015-1687-4>
- [12] R. B. Handfield and E. L. Nichols, *Introduction to Supply Chain Management*. New Jersey, 1999.
- [13] K. K. Castillo-Villar, "Metaheuristic Algorithms Applied to Bioenergy Supply Chain Problems: Theory, Review, Challenges, and Future," *Energies*, vol. 7, no. 11, pp. 7640–7672, 2014.
<http://dx.doi.org/10.3390/en7117640>
- [14] S. K. Kumar, M. K. Tiwari, and R. F. Babiceanu, "Minimisation of supply chain cost with embedded risk using computational intelligence approaches," *International Journal of Production Research*, vol. 48, no. 13, pp. 3717–3739, Jul. 2010.
<http://dx.doi.org/10.1080/00207540902893425>
- [15] H. Ramalhinho Dias Lourenço, "Supply chain management: an opportunity for metaheuristics," *UPF Economics and Business Working Paper*, no. 538, 2001.
- [16] H. Min and G. Zhou, "Supply chain modeling: past, present and future," *Computers & industrial engineering*, vol. 43, no. 1, pp. 231–249, 2002.
[http://dx.doi.org/10.1016/S0360-8352\(02\)00066-9](http://dx.doi.org/10.1016/S0360-8352(02)00066-9)
- [17] F. Glover, "Tabu search-part I," *ORSA Journal on computing*, vol. 1, no. 3, pp. 190–206, 1989.
<http://dx.doi.org/10.1287/ijoc.1.3.190>