International Journal of Mechanical, Industrial and Aerospace Sciences
ISSN: 2517-9950
Vol:9, No:5, 2015

Symbiotic Organism Search (SOS) for Solving the Capacitated Vehicle Routing Problem

Ruskartina Eki, Vincent F. Yu, Santosa Budi, A. A. N. Perwira Redi

***Abstract*—**This paper introduces symbiotic organism search (SOS) for solving capacitated vehicle routing problem (CVRP). SOS is a new approach in metaheuristics fields and never been used to solve discrete problems. A sophisticated decoding method to deal with a discrete problem setting in CVRP is applied using the basic symbiotic organism search (SOS) framework. The performance of the algorithm was evaluated on a set of benchmark instances and compared results with best known solution. The computational results show that the proposed algorithm can produce good solution as a preliminary testing. These results indicated that the proposed SOS can be applied as an alternative to solve the capacitated vehicle routing problem.

***Keywords*—**Symbiotic organism search, vehicle routing problem, metaheuristics, Solution Representation.

NOTATION

n	: set of coordinate customers
m	: set of coordinate of vehicle
ub	: upper bound
lb	: lower bound
I_{size}	: Size of Organism (ecosystem)
eco_size	: ecosystem size
it	: Iteration
max_it	: Maximum iteration
N	: Non Improvement count
X_{inew}	: X_i new (result) from particular phase
PF	: Parasite Force
BF	: Beneficial factor
x_{id}	: Position of the ith organism at the dth dimensions
R_{ij}	: Route of the jth vehicle corresponding to the ith organism

I. INTRODUCTION

CAPACITATED vehicle routing problem (CVRP) or Vehicle Routing Problem (VRP) was first introduced by [1]. It had been proven by [2] that this problem is NP-hard problem since it is not solvable in polynomial time. CVRP is one of the most widely-studied problems in combinatorial optimization and the literature provides an extensive stream of heuristics and metaheuristics solution techniques. CVRP can be briefly described as a set of n customers that must be served by m number of homogeneous vehicles. Let $G = (V,A)$ be a complete graph where $V = \{0,\dots,n\}$ is the vertex set and A is arc set. Vertex $i=1,\dots,n$ correspond to the customer and vertex 0 correspond to the depot. Each customer must be assigned to exactly one vehicle to be visited at exactly once. At each visit, vehicle capacity Q and maximum allowable route length L should not be violated. Each customer i is associated with a non-negative demand d_i, service time s_i, and travel cost c_{ij}. The objective is to satisfy total demand of customers while minimizing total network cost [3].

Several classical heuristics approaches were proposed to solve capacitated vehicle routing problem (CVRP). Clarke and Wright [4] used savings algorithm to solve CVRP. This algorithm usually applied to CVRP that emphasizes number of vehicle as decision variables. Sequential insertion heuristics was proposed by [5] and [6] to solve CVRP with undefined number of vehicles. Gillet and Miller [7], Wren and Holliday [8] presented elementary clustering method called as sweep algorithm; and [9] proposed generalized assignment-based algorithm and [10] proposed location-based heuristics. These last two heuristics assume that the number of vehicles is fixed. In recent years, several metaheuristics algorithm have been proposed for CVRP. Robuste, Daganzo, and Souleyrette [11] implemented simulated annealing with neighborhood structure in the context of VRP, the algorithm was tested on several instances but no comparisons are available to verify the performance. Then, [12] proposed a Simulated Annealing with Tabu Search algorithm and were more successful in his implementation. Some examples of other metaheuristics were also proposed such as Genetic algorithms [13], [14]; Ant colony optimization [15], Artificial Bee Colony algorithm [16], and Particle Swarm Optimization [17].

Cheng and Prayogo [18] presented a new metaheuristics called Symbiotic Organism Search (SOS) inspired from interaction among organisms in the ecosystem known as symbiotic relationship. This algorithm was proposed initially to solve continuous engineering optimization problems. SOS showed considerable robustness in its performance when tested on complex mathematical benchmark problems. SOS has never been used to solve discrete problem, such as, routing, scheduling, or assignment problems. This could be our motivation to first introduce SOS to solve discrete problem. In this paper, we introduce SOS to solve CVRP. We evaluate the performance of the SOS using classical benchmark instances. In order to enhance the solution quality, we adopted the use of solution representation 1 (SR-1) presented by [19]. Originally SR-1 is used for Particle Swarm Optimization (PSO) to deal with CVRP. Solution representation is defined as an encode solution for each particle or candidate solution, while the method to transform it to problem specific solution is called decoding method. This solution representation is constructed with original PSO framework which uses real-valued particle positions instead of discrete-valued representation. The

Vincent F. Yu, Ruskartina Eki and A. A. N. Perwira Redi are with the National Taiwan University of Science and Technology, Taiwan, ROC. (phone: +886975181046; fax: +886227376344; e-mail: ekiruskartina@gmail.com).
Santosa. Budi, and Ruskartina. Eki are with Sepuluh Nopember Institute of Technology, Surabaya, Indonesia (e-mail: shima1907@yahoo.com).

preliminary test runs show that SOS algorithm with SR-1 can obtain promising results for the CVRP.

The remainder of this paper is organized as follows: Section II describes in detail the symbiotic organism search algorithm. Section III discusses SOS with SR-1. Section IV presents the computational results. Finally, section V gives the conclusions and directions for future researches.

II. Symbiotic Organism Search Algorithm

Almost all metaheuristics algorithm were inspired from natural biological phenomena. Particle swarm optimization simulates the social behavior of bird flocks. Genetic algorithm, a type of evolutionary algorithm mimics the process of natural selection [20]. SOS simulates interactions between two organisms survival in the ecosystem. This interaction is also known as Symbiosis. Symbiotic relationship is defined as reliance-based relationship among organism in order to fulfill their sustenance or even survive in nature. Generally, there are three kinds of symbiotic relationships namely, mutualism, commensalism, and parasitism. In mutualism, organisms benefits from each other. Like for example, the interaction between starlings and buffalo. Starlings get ticks from buffalo's skin for sustenance. The itching on buffalo's skin will be reduced in return. Commensalism takes place when an organism gets benefits while the other is not significantly harmed and helped. Like for the example, the interactions between remora fish and shark. Remora fish eats leftovers from the shark without bothering the shark at all. Parasitism takes place when an organism obtains benefits from the interaction while other is harmed. Like for example, the anopheles mosquito and human body. Anopheles inducts parasite into the human body which poses fatal threats causing the body to eventually die. Generally, most organisms in ecosystem are doing these kinds of symbiotic relationships to adapt to environmental changes and to create survival strategies over long periods of times [18].

Like other population-based metaheuristics, SOS utilizes an initial population called ecosystem in order to provide candidate solutions in the searching space to obtain optimal solution. Organisms are generated randomly from the searching space that has upper and lower bounds. Then, the best solution (X_{best}) is selected among all organisms. Commonly, metaheuristics have operators in order to generate a new solution in each iteration. The phases in SOS such as mutualism, commensalism, and parasitism serve as the operators. Each organism interacts with other organisms randomly in the population through all phases. For example, with an *eco_size* of *n* member organisms, each member will go through each phase of the algorithm. This is the inner loop of the algorithm. Then, the process repeats until the termination criteria is reached. This is the outer loop of the algorithm specifically using maximum number of iterations as a stopping policy. Fig. 1 describes the simple SOS algorithm procedure.

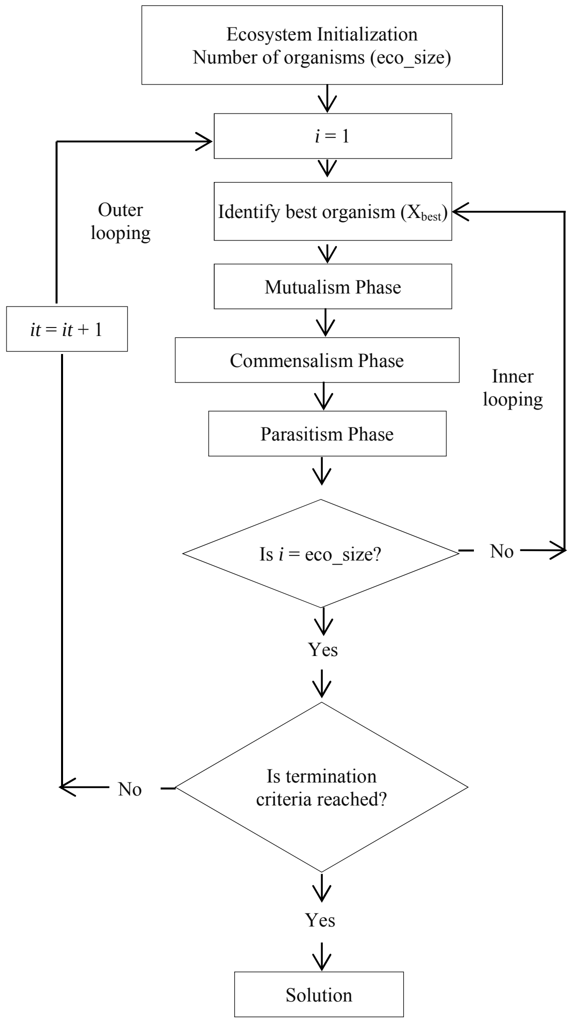

Fig. 1 SOS Algorithm Procedure

III. SOS Framework for Solving CVRP

The proposed algorithm to solve CVRP is based on SOS proposed by [18] and added with solution representation 1 proposed by [19].This algorithm uses four parameters, namely I_{size}, *max_it*, *Non_improve*, and *Parasite force* (PF). I_{size} is the number of organisms in the ecosystem usually called population size. *Max_it* is maximum number of iteration and *Non_improve* is the allowable number of update organisms during which X_{best} has not been improved. *Parasite force* is a parameter from 0 to 1 that represents the probability of the creation of *Parasite_vector*.

Algorithm 1 presents the proposed SOS algorithm. It begins with generating I_{size} of organisms with a particular representation that corresponds to I_{size} different sets of vehicles. Then, we select the best organism from the ecosystem to be the X_{best} . Then, a movement procedure which is called update organism is applied to each organism. Whenever a better set of route is found after all phase done, X_{best} is updated. In addition, in order to improve the solution quality, we apply local search improvement using neighborhood operator such as swap, reverse, and insert to each organism. The algorithm is terminated when the best solution (X_{best}) has not been improved in *Non_improve* or the iteration reached *max_it*.

```
Main code
Input   : I_size, max_it, Non_improve, PF,ub,lb
Output  : X_best, F_best

Begin
  Step 1 : Generate X_i , i= 1,2,3, . . . ., I_size in the range [ub, lb]
           Set it=0 & N=0
  Step 2 : For i= 1 to I_size
           Step 2.1 : Calculate objective of X_i (F_best)
           Step 2.2 : Identify best organism (X_best)
                      If obj (X_i) < obj (X_best) ,
                      Update X_best = X_i
  Step 3 :Update organism
           For i = 1 to I_size
           Step 3.1 :Mutualism phase
           Step 3.2 :Commensalism phase
           Step 3.3 :Parasitism phase
         Go to step 2
  Step 4 : Local search improvement
         Go to step 2
  Step 5 : if N=Non_improve or it=itmax
           {Terminate SOS Procedure}
            Else : Go to step 3
  Step 6 : return X_best
End
```

Algorithm 1. SOS framework for CVRP

A. Mutualism Phase

Two organisms interact in mutualism phase. X_i is an organism selected sequentially while X_j is selected randomly from the ecosystem. Both organisms interact with the purpose of increasing mutual survival advantage in the ecosystem. The relationship is represented by Mutual_vector equation.

Mutual_vector represents the relationship characteristics between organism X_i and X_j. BF_1 and BF_2 represent the beneficial factor from both organisms. The value of BF_1 and BF_2 are determined randomly as either 1 or 2. Then new candidate solution from X_i and X_j are calculated as in step 1.3 in Algorithm 2. Organisms are updated if their fitness is better than their previous. Finally, only the fittest organism is selected to go to the next phase. By doing this phase, we update two organisms at once. The pseudocode for mutualism phase is presented in Algorithm 2.

B. Commensalism Phase

The fittest X_i from mutualism phase becomes the input for commensalism phase. Similar to Mutualism phase, an organism X_j is selected randomly from the ecosystem. Then the interaction between X_i and X_j is calculated as step 1.2 in Algorithm 3. Different from Mutualism phase, output of commensalism is only one organism (X_i). X_i is updated only if its new fitness is better than previous. The pseudocode of Commensalism is shown in Algorithm 3.

```
Input    : X_i,X_j, j=1,2,3 . . . .I_size
Output  : X_i ,X_j
Begin
  Step 1 :
    Step 1.1 : Select X_j randomly from ecosystem and X_j ≠ X_i
    Step 1.2 : Calculate mutual vector and Beneficial factor
           Mutual Vector = (X_i + X_j)/2
           BF_1 & BF_2 = random either 1 or 2
    Step 1.3 : Calculate X_inew and X_jnew
           X_inew= X_i + rand(0,1)*(X_best – Mutual_Vector*BF_1)
           X_jnew = X_j + rand(0,1)*(X_best – Mutual_Vector*BF_2)
  Step 2 :Decode X_inew dan X_jnew
  Step 3 : Calculate objective X_inew and X_jnew
  Step 4 : if obj (X_inew) <obj (X_i) && obj (X_jnew) < (X_j)
            Update  X_i = X_inew & X_j = X_jnew
  Step 5 : return X_i & X_j
End
```

Algortihm 2. Pseudocode for Mutualism Phase

```
Input    : X_i,X_j, j=1,2,3 . . . .I_size
Output  : X_i
Begin
  Step 1 :
    Step 1.1 : Select X_j randomly from ecosystem and X_j ≠ X_i
    Step 1.2 : Calculate X_inew
           X_inew = X_i + rand(-1,1)*(X_best– X_j)
  Step 2 :Decode X_inew
  Step 3 : Calculate objective X_inew
  Step 4 : if obj (X_inew) <obj (X_i)
           Update  X_i = X_inew
  Step 5 : return X_i
End
```

Algortihm 3. Pseudocode for Commensalism Phase

C. Parasitism Phase

Parasitism phase has the same structure with mutation operator in Genetic Algorithm. First, X_i from commensalism phase becomes the input for parasitism phase. From X_i, we create an artificial parasite called "Parasite_vector". Parasite_vector is created by duplicating organism X_i, then modifying it based on the Parasite force.

Then, X_j is selected from the ecosystem to interact with *Parasite_vector*. If X_j is better than the Parasite_vector, then X_i will survive. Otherwise, *Parasite_vector* kills X_j and replaces its current position. The steps of this phase are shown in Algorithm 4.

D. Solution Representation and Decoding Method

The solution representation 1 or SR-1 of CVRP consists of (n+2m) dimension. The first n dimensions refer to customers and the last 2m dimensions are related to vehicle. In each phase, the decoding method is applied after the organism is updated. Each organism is encoded as a real number. The notations and decoding algorithm for SR-1 are presented in Algorithm 5.

```
Input   : X_i,X_j, j=1,2,3 . . . .I_size
Output  : X_j
Begin
    Step 1 : Select X_j randomly from ecosystem and X_j ≠ X_i
    Step 2 :Generate parasite vector (X_Par_Vec) from X_i
        Step 2.1 : Generate r = random (0,1)
        Step 2.2 : If r < PF
                Mutate X_i using random number with a range
                [ub,lb]
    Step 3 :Decode X_Par_Vec and X_j
    Step 4 : Calculate objective X_Par_Vec and X_j
    Step 5 : if obj (X_Par_Vec) <obj (X_j)
            Update X_j= X_Par_Vec
    Step 6 : return X_j
End
Go to Step 2 (Main Code)
End
```

Algorithm 4. Pseudocode for Parasitism Phase

IV. Computational Results

The SOS algorithm is implemented in C++ language using Microsoft visual studio, on a PC with Intel Core 2 Duo CPU, 3 Ghz, and 4 GB RAM. The SOS parameters are as follows: number of organism (I_{size}) = 100, number of iteration (*it*) = 100, *Non-improvement* = 50, and *Parasite Force* = 0.8.

The characteristics of SOS have some resemblance with PSO algorithm. The movement mechanism of PSO called velocity has the same structure with mutualism phase of SOS. Based on this reasons, the improvement of SOS can follow how PSO algorithm is improved to solve CVRP. By the same reason, SR-1 applied to improve PSO for CVRP is proposed in SOS algorithm.

The computational experiment is conducted on the Christofides et al (1979) benchmark problem. This benchmark have 14 instances and each cluster of instances have different number of customers (n), number of vehicles (m), maximum capacity constraint (Q), maximum routing time (L) and service time for each customers (s). Table I lists the characteristics of 14 instances and provides information about the best solution obtained by the proposed SOS.

Instances with 50 – 100 number of customers show better results with difference ranging from 0 – 2.99%. Furthermore, instances with 120 - 199 numbers of customers show differences ranging from 3.11 – 11.34. The result from SOS, especially for the large instances still needs further improvement. However, the average deviation for all instances is relatively small at 3.29%. In addition, the result for vrpnc1 is exactly the same with best known solution. With this preliminary test, SOS for CVRP shows promising results.

V. Conclusion and Future Study

Symbiotic organism search for solving capacitated vehicle routing problem is presented in this paper. The SOS was initially proposed to solve engineering optimization problem, a continuous mathematical problem. This paper proposed new approach of SOS to solve the discrete problem, CVRP. SOS is implemented for CVRP with Solution representation 1 decoding method. The computational result shows that with the preliminary tests, SOS can obtain reasonable and promising results.

Future researches may focus on developing SOS for solving CVRP that can handle both small and large instances efficiently. In addition, the researcher may also consider using SOS to solve another discrete problem.

```
Input  : X_i,  i = 1,2,3 . . . . I_size
Output :
Begin
    Step 1 : Construct the priority list of customer (U)
        Step 1.1 : Set S = {1,2,…,n} and U = ø
        Step 1.2 : Select c from set S where x_ic = min_{dϵS} x_id
        Step 1.3 : Add c to the last position in set U
        Step 1.4 : Remove c from set S
        Step 1.5 : Repeat step 1.2 – 1.4 until S=ø
    Step 2 : Construct vehicle priority matrix (V)
        Step 2.1 : Set the vehicle reference position. For j=1…m, set
                   x_refj = x_{i,n+j} and y_ref = x_{i,n+m+j}
        Step 2.2 : For each customer k, k=1….n.
            Step 2.2i  : For each vehicle j=1…m, set δ_j as the
                         Euclidean distance between customer k and
                         the reference point of vehicle j.
            Step 2.2ii  : Build set S = {1,2,…m} and V_k= 0
            Step 2.2iii : Select c from set S where δ_c = min_{dϵS} δ_d
            Step 2.2iv  : Add c to the last position in set V_k
            Step 2.2v   : Remove c from set S
            Step 2.2vi  : Repeat step 2.2iii – 2.2v until S = ø
    Step 3 : Construct vehicle route
        Step 3.1 :Set k = 1
        Step 3.2 : Add customer one by one to the route
            Step 3.2i    : Set l = U_k and p = 1
            Step 3.2.ii  : Set j = V_{l,p}
            Step 3.2iii  : Make a candidate of new route by inserting
                           customer l to the best sequence in the
                           route R_ij, which has the smallest
                           additional cost.
            Step 3.2iv   : Check the capacity and route time
                           constraint of the candidate route.
            Step 3.2v    : if a feasible solution is reached, update the
                           Route R_ij, with the candidate route, then
                           apply 2-opt procedure to the route R_ij;
                           go to step 3.3.
            Step 3.2vi   : if p=m, go to step 3.3. otherwise, set p =
                           p+1 and go to step 3.2ii
            Step 3.3 : if k=n, stop. Otherwise, set k=k+1 and repeat
                       3.2
End
```

Algorithm 5. Solution representation 1 decoding method [19]

TABLE I
COMPUTATIONAL RESULT OF SOS FOR CVRP

Instance	n	m	Q	s	L	BKS*	SOS best	Diff (%)	Comp. Time (m)
vrpnc1	50	5	160	0	∞	524.61	***524.61***	0.00	0.35
vrpnc2	75	10	140	0	∞	835.26	842.39	0.85	2.28
vrpnc3	100	8	200	0	∞	826.14	839.70	1.64	2.31
vrpnc4	150	12	200	0	∞	1028.42	1060.38	3.11	18.16
vrpnc5	199	17	200	0	∞	1291.29	1378.83	6.78	46.36
vrpnc6	50	6	160	10	200	555.43	557.89	0.44	0.49
vrpnc7	75	11	140	10	160	909.68	936.85	2.99	1.60
vrpnc8	100	9	200	10	230	865.94	872.29	0.73	3.16
vrpnc9	150	14	200	10	200	1162.55	1294.37	11.34	14.44
vrpnc10	199	18	200	10	200	1395.85	1494.66	7.08	39.83
vrpnc11	120	7	200	0	∞	1042.11	1047.83	0.55	6.95
vrpnc12	100	10	200	0	∞	819.56	820.25	0.08	6.68
vrpnc13	120	11	200	50	720	1541.14	1658.00	7.58	10.19
vrpnc14	100	11	200	90	1040	866.37	891.66	2.92	6.43
Average								3.29	

*Best Known Solution; Diff = (BKS – SOS Best) / BKS * 100%

REFERENCES

[1] G. B. Dantzig., J. H. Ramser. *The truck dispatching problem*. Management Science, 6:80, 1959.

[2] Lenstra, J.K., Kan. A.H.G.R. *Complexity of vehicle routing and scheduling problems*. Networks 11, 221 – 227. 1981.

[3] Toth. P, Vigo. D. *The vehicle routing problem. SIAM Monograph on Discrete Mathematics & Application*. Philadelphia, PA. 2002

[4] G. Clarke, J.V. Wright. *Scheduling of vehicle from a central depot to a number of delivery points*. Operation research, 12:568-581, 1964

[5] R.H. Mole, S.R. Jameson. *A sequential route-building algorithm employing a generalized saving criterion*. Operational Research Quarterly, 27:503-511, 1976.

[6] N. Christofides, A. Mingozzi, P. Toth. *The vehicle routing problem.* In N. Christofides, A. Mingozzi, P. Toth, C. Sandi, editors, *Combinatorial Optimization*. Wiley, Chichester, UK, 1979, pp 315-338

[7] B. E. Gillett, L.R. Miller. *A heuristic algorithm for vehicle dispatch problem*. Operation research, 22:340-349, 1976.

[8] A. Wren, A. Holliday. *Computer scheduling of vehicles from one or more depots to a number of delivery points*. Operational Research Quarterly, 23:333-334, 1972

[9] M. L. Fisher, R. Jaikumar. *A generalized assignment heuristic for vehicle routing problem*. Networks, 11:109-124, 1981.

[10] J. Bramel, D. Simche-Levi. *A location based heuristic for general routing problems*. Operation research, 43:649-660, 1995.

[11] F. Robuste, C. F. Daganzo, R. Souleyrette. *Implementing vehicle routing models*. Transportation Research B, 24:263-286, 1990.

[12] I.H. Osman. *Metastrategy simulated annealing and tabu search algorithm for the vehicle routing problem.* Annals of Operations Research, 41:421-451, 1993.

[13] Baker. B. M, Ayechew, M.A. *A genetic algorithm for vehicle routing problem*. Computer & Operation Research 30:787-800, 2003.

[14] Berger. J, Barkoui. M. *A new hybrid genetic algorithm for the capacitated vehicle routing problem*. Journal of the Operational Research Society 54:1254-1262, 2003.

[15] Mazzeo. S, Loiseau. Irene. *An ant colony algorithm for the capacitated vehicle routing.* Electronic Notes in Discrete Mathematics, 18:181-186, 2004.

[16] Szeto. W. Y, Yongzhong Wu, Sin. C. Ho. *An artificial bee colony algorithm for the capacitated routing problem*. European Journal of Operational Research 215:126-135, 2011.

[17] Ai. T. J, Kachitvichyanukul. V. *A particle swarm optimization for capacitated vehicle routing problem*. International Journal of Logistics and SCM Systems, 2:50-55, 2007.

[18] Cheng, M. Y, Prayogo, D. *Symbiotic organism search: A new metaheuristic optimization*. Computers and Structures, 139:98-112, 2014.

[19] Ai. T. J, Kachitvichyanukul. V. *Particle swarm optimization and two solution representations for solving the capacitated vehicle routing problem*. Computers & Industrial Engineering, 56:380-387, 2009.

[20] Talbi. El-Ghazali. *Metaheuristics from design to implementation*. Wiley, University of Lille – CNRS – Inria. 2009.

[21] Santosa. Budi, Willy. Paul. *Metoda Metaheuristik Konsep dan Implementasi*. Surabaya, Indonesia. 2011.