A Hybrid Self-adaptive Global Best Harmony Search Algorithm for the Vehicle Routing Problem with Time Windows

Fateme Maleki
Department of Computer Science, Faculty of Mathematics, University of Sistan and Baluchestan, Zahedan, Iran
P.O.Box 98135-674, Tel. +98 54 3113 2500

Majid Yousefikhoshbakht
Assistant Professor, Department of Mathematics, Faculty of Science, Bu-Ali Sina University, District 2, Hamadan, +98 81 3838 1601
Hamadān Province, Iran
khoshbakht@basu.ac.ir

Amin Rahati
Department of Computer Science, Faculty of Mathematics, University of Sistan and Baluchestan, Zahedan, Iran
P.O.Box 98135-674, Tel. +98 54 3113 2500

Abstract

This paper presents a hybrid self-adaptive global best harmony search algorithm (HSGHSA) to improve the performance of harmony search algorithm (HSA) for solving vehicle routing problems with time windows (VRPTW). To explore the search space more efficiently, the proposed HSGHSA couples an improved variant of HSA called global best harmony search algorithm with a self-adaptive mechanism for tuning its control parameters. Moreover, the HSGHSA adopts six local search (LS) neighborhood structures to enhance its exploitation capability. The effectiveness of HSGHSA is evaluated against Solomon's VRPTW benchmark and its performance is compared with HSA and several state-of-the-art algorithms. The obtained results confirm that the HSGHSA produces very competitive results compared to the other algorithms.

Keywords: Harmony search algorithm, Global best harmony search algorithm, Vehicle routing problem, Time windows, Metaheuristic algorithms.

1. Introduction

Transportation is an important domain of human activity that supports and makes possible most other social and economic activities. Transportation by vehicle is one of the most common approach to accomplish such a process. However, finding an optimum transportation route is a complex problem that has been widely investigated for the distribution systems. The problem known as the vehicle routing problem (VRP) is a combinatorial optimization problem and belongs to the family of NP-hard problems (Fu, Aloulou, & Triki, 2017). There are various types of VRPs. For example, the VRP which comes with the time window constraints is called VRP with Time Windows (VRPTW). The problem is considered closer to real life situations due to its time window constraints in comparison to capacitated VRP in which only vehicle capacity constraint is considered. In VRPTW, we have the task to provide goods of a specified quantity to a set of geographically scattered customers. Each customer has a specified time window within which they can be served. We have set of vehicles having limited capacity at our disposal to provide service. The aim is designing a least cost routing plan to deliver the goods from the depot to the customers under the following conditions:

- Each vehicle must start its route from the depot and end it at the depot.
- The total demands of all customers in each route should not exceed the capacity of the vehicle.

- If a vehicle arrives earlier than the start time window of a customer, it must wait to serve the customer during its time window.
- If the vehicle arrives after the end time window of that customer, then it cannot serve the customer.
- Each customer must be served only once during his/her time window.

So far, many researchers from various fields have tried to solve the VRPTW using exact, heuristic, and metaheuristic methods. Exact methods are able to find optimal solutions for VRPTW but they are only recommended for problems with small size VRPTW. In contrast, heuristic, and metaheuristic algorithms are able to tackle large-sized VRPTW problems but these methods do not guarantee to obtain the optimal solutions (Bräysy & Gendreau, 2005a).

Azi et al. introduced an exact algorithm for a VRPTW (Azi, Gendreau, & Potvin, 2010), but Yassen et al. (Yassen et.al., 2015a) proposed a meta-harmony search algorithm (meta-HSA) to solve VRPTW which uses two HS algorithms, an HSA-optimizer and HSA-solver. The HSA-optimizer adaptively adjusts the components and the configurations of the HSA-solver (LS) based on the search status. The HSA-solver, which is a hybridization of HSA and local Search, takes the configuration generated by the HSA-optimizer as input and then solves the given problem instance. The proposed meta-HSA in this paper is used to adjust the parameter values, LS types and LS configurations (parameter values and neighborhood operators). Yassen et al. (Yassen et.al., 2015b) applied a hybrid metaheuristic algorithm to solve VRPTW, which hybridizes HSA with simulated annealing for improving the performance of HSA. They also investigated the effect of hybridizing LS algorithms with HSA for VRPTW in (Yassen et. al., 2015c) and used three well-known LS algorithms: hill climbing, simulated annealing, and reactive tabu search to hybridize with the harmony search algorithm. Ursani et al. (Ursani et. al., 2011) presented localized optimization framework in which a problem is decomposed into sub-problems and optimization is done on those sub-problems independent of each other. They used a genetic algorithm as an optimization method and adapted it to fit within the framework of VRPTW as a problem domain. Nagata et al. (Nagata, Bräysy, & Dullaert, 2010) developed a penalty-based memetic algorithm for the VRPTW by extending the edge assembly memetic algorithm (EAMA) of Nagata (Nagata, 2007) for the capacitated VRP. The suggested EAMA is based on the two-stage approach. An initial population of solutions, each consisting of the same number of routes, is generated with a sophisticated route minimization procedure developed by Nagata and Braysy (Nagata & Bräysy, 2009). A subsequent procedure of the EAMA is then applied for minimizing the total travel distance for the determined number of routes. In this paper, adapted edge assembly crossover (EAX) (Nagata, 2006) is used to the VRPTW and a novel penalty function for the time window violation is developed. Yu et al. (Yu , Yang , & Yao, 2011) proposed a hybrid approach, which consists of ant colony optimization (ACO) and tabu search, to solve the VRPTW. Baños et al. (Baños et. al., 2013) proposed a multi-objective procedure based on simulated annealing called the multiple temperature Pareto simulated annealing (MT-PSA). This paper deals with a multi-objective variant of the VRPTW that simultaneously minimizes the travelled distance and the imbalance of the routes. This imbalance is analyzed from two perspectives: the imbalance in the distances travelled by the vehicles, and the imbalance in the loads delivered by them. Ding et al. (Ding et. al., 2012) presented a hybrid ant colony optimization to solve VRPTW by adjusting pheromone approach and introducing a disaster operator. This modification leads to prevent the search process from getting trapped in the local optimal solution by taking the candidate list into consideration and combining the ACO with the saving algorithm and ƛ-interchange mechanism. In this work, the saving algorithm (Clarke & Wright, 1964) is used to initially assign each customer to a separate route and ƛ -interchange mechanism (Osman, 1993) to improve the convergence speed of ACO. Table 1 shows a summary of the history of VRPTW.

Table 1. The summary of the history of VRPTW

Author	The proposed approach	The number of best known solutions obtained	Year
Azi et. al. (2010)	The introduction of a branch-and-price approach	Not available	2010
Nagata et al. (2010)	Penalty-based memetic algorithm for the VRPTW by extending the edge assembly memetic algorithm (EAMA)	Obtains 45 best known solutions and improves 1 than the best-known solutions in the literature	2010
Ursani et al. (2011)	Localized Genetic Algorithm (LGA)	Obtained 18 best known solutions and 10 new best solutions	2011
Yu et al. (2011)	A hybrid of ant colony optimization and Tabu search (ACO–Tabu)	Obtains 41 best known solutions and improves 4 than the best-known solutions in the literature	2011
Ding et al. (2012)	Hybrid ant colony optimization (HACO)	Obtains 3 best known solutions and improves 3 than the best-known solutions in the literature	2012
Baños et al. (2013)	Multi-objective procedure based on simulated annealing, the multiple temperature pareto simulated annealing (MT-PSA)	Not available	2013
Barbucha (2014)	Cooperative Population Learning Algorithm (CPLA)	Obtains 17 best known solutions and improves 9 than the best-known solutions in the literature	2014
Yassen et al. (2015a)	Meta-harmony search algorithm (meta-HSA)	Not available	2015

Harmony search algorithm (HSA) introduced for the first time in 2001 (Geem et. al., 2001) was inspired by the process of the orchestra playing the music. In comparison to other main stream meta-heuristics, HSA has fewer mathematical parameters and is robust enough for the wide range of applications. That is why HSA gained much attention for solving engineering problems (Lin et. al., 2017; Gao et. al., 2015). However, the performance of HSA highly depends on parameter selection and inaccurate parameter selection can lead to poor performance of HSA in local searches. On the other hand, population-based methods are integrated with LS algorithm (Blum et. al., 2011) to increase the exploitation process. Therefore, we hybridize HSA with LS algorithms in order to improve the quality of solutions and enhance the exploitation in this work. Furthermore, we used hill climbing (HC), simulated annealing (SA) and great deluge algorithm (GD) as the LS algorithms randomly mixed with self-adaptive global best harmony search algorithm (SGHSA) in the proposed HSGHSA algorithm. In addition, six local search neighborhood structures i.e., relocate, interchange, end customer interchange, or-opt, 2-opt star, and cross exchanges (Bräysy & Gendreau, 2005a) are also used for further improvement in the solution. The Solomon's VRPTW benchmark is employed to evaluate the performance of the HSGHSA compared to the results of the standard HSA and the best-known results in the literature. The results show that the proposed algorithm is significantly efficient and finds closely the best-known solutions (BKSs) for most of the instances.

The rest of the paper is structured as follows. In the Section 2, classic HSA and SGHSA are explained. The proposed method is described in more detail in the Section 3 and in the next Section, the experimental design is presented and the proposed algorithm is compared with some of the famous metaheuristic algorithms on standard VRPTW problems in the Section 5. Finally, the conclusions are presented in the Section 6.

2. Harmony Search Algorithm

HSA is a new population-based meta-heuristic algorithm inspired by the process of playing the music when a musician is looking for better harmony. In the algorithm, harmony in music is similar to solution vector, and the musician's improvisations are similar to local and global search schemes. HSA is inclusive of three basic phases, namely, initialization, improvisation of a harmony vector and updating the harmony memory (HM). In the initialization phase, an initial population of harmonies is randomly created and stored in a HM. In the next phase, a new harmony by using a memory consideration rule and a pitch adjustment rule are improvised. In the final phase, the harmony memory is updated if the new harmony is better than the worst harmony vector in the HM. Generally, HSA has five parameters as follows:

- Harmony memory size (HMS) is the number of solution vectors in the HM.
- Harmony memory consideration rate (HMCR) is used to improvise a new harmony and varies between 0 and 1. Depending on the HMCR, the decision variable of the new harmony is selected for HM.
- Pitch adjustment rate (PAR) is the value, which varies between 0 and 1. The decision variable of new harmony is regulated if the randomly generated number between 0 and 1 turns out to be less than the value of PAR.
- The step size of the PAR parameter called distance bandwidth (BW).
- The termination condition based on the number of improvisations (NI).

2.1. Self-adaptive global best harmony search algorithm

The self-adaptive global best harmony search algorithm (SGHSA) (Quan-Ke Pan et. al., 2010) is a variant of HSA used an adaptive parameter tuning method and a new improvisation scheme. This algorithm uses a fixed user-specified value of the harmony memory size (HMS) and the number of improvisations (NI). HMCR and the pitch adjustment rate (PAR) parameters are dynamically adapted to a suitable range by recording their historic values corresponding to created harmonies in the harmony memory (HM). SGHSA starts with specified standard deviation and initial values of HMCR and PAR equivalent to their normal distribution means HMCRm and PARm. During the evolution, if the generated new harmony is replaced with the worst harmony in the HM then HMCR and PAR values are recorded. After a specified number of iterations learning period (LP), HMCRm and PARm are recalculated by averaging all the recorded HMCR and PAR values during this period. SGHSA consists of the following five steps.

Step 1: Problem and algorithm parameter initialization.

- Set parameters HMS, LP, NI, $HMCR_{std}$ and PAR_{std}
- Initialize BWmax, BWmin, HMCRm and PARm

Step 2: Harmony memory initialization and evaluation.

- Initialize HM from a uniform distribution and evaluate it. Set the generation counter lp=1.

Step 3: New harmony improvisation.

- Generate HMCR and PAR according to HMCRm and PARm. Yield BW according to BWmax and Bwmin.
- Improvise a new harmony: To generate a new harmony, a new decision variable (with probability HMCR) from the best harmony is selected in the HM or randomly generate (with probability 1-HMCR). If the new decision variable has been inherited from the best harmony in the HM, then it should be adjusted (with probability PAR).

Step 4: Harmony memory update.

- Updated HM if the new harmony is better than the worst harmony in the HM and record the values of HMCR and PAR.

Step 5: Termination criterion check.

- If lp=LP, recalculate HMCRm and PARm according to the recorded values of HMCR and PAR. Reset lp=1; otherwise increase generation counter lp. If NI is completed, return the best harmony vector in the HM; otherwise go back to step 3.

Figure 1 shows the pseudo code of the SGHSA described above.

```
Step 1: Set parameters HMS, LP and NI.
Step 2: Initialize $BW_{max}$, $BW_{min}$, HMCRm and PARm.
Step 3: Initialize and evaluate HM. Set generation counter lp = 1.
Step 4: Generate HMCR and PAR according to HMCRm and PARm. Yield BW according to
$BW_{max}$ and $BW_{min}$.
Step 5: Improvise a new harmony Xnew as follows:
    for (j = 1 to D) do
        if $(rand < HMCR)$ then
            $x_{new}(j) = x_a(j) \pm rand \times BW$
            if $(rand < PAR)$ then
                $x_{new}(j) = x_B(j)$
            endif
        else
            $x_{new}(j) = LB(j) + rand \times (UB(j) - LB(j))$
        endif
    end for
Step 6: If $f(X_{new}) < f(X_w)$, Update the HM as $X_w = X_{new}$ and record the values of HCMR and
PAR.
Step 7: If lp = LP, recalculate HMCRm (PARm) according to the recorded values of HCMR (PAR)
and reset lp = 1; otherwise,lp = lp + 1;
Step 8: If NI is completed, return the best harmony vector XB in the HM; otherwise go back to
step 4.
```

Figure 1. The pseudo-code of SGHSA

3. The solution method

In this section, the proposed algorithm is explained in more details. This work aims to propose a method that is better than all proposed methods based on harmony search for VRPTW in the literature by comparing the results of the application of HSGHSA with those obtained via the state-of-the-art methods.

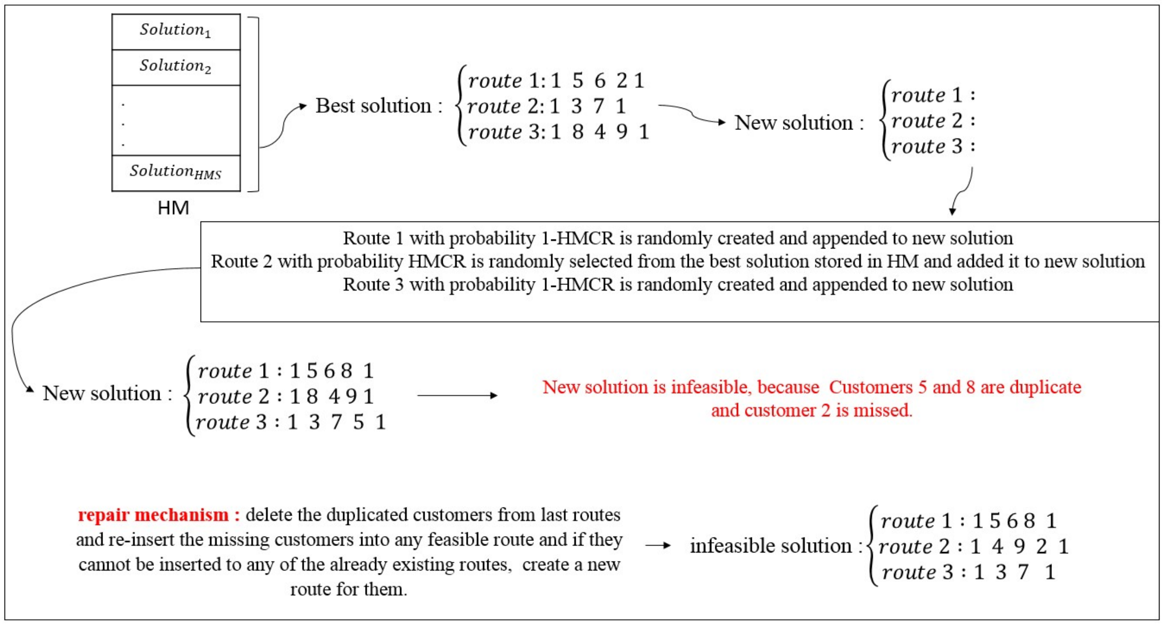
3.1. Adapting SGHSA to solve the VRPTW

The SGHSA starts with a population of harmonies and, iteratively, generates a new harmonious solution for a specific problem instance. This algorithm process consists of five phases as follows:

1. Initialize the problem and algorithm parameters.
2. Initialize the harmony memory and randomly generate the initial population of solutions stored in HM: To create a solution first, an empty route is created and an un-routed customer is randomly selected and added to the current route such that it does not violate the VRPTW constraints. Then, the nearest un-routed customer to the last inserted customer is selected and appended to the current route (if it satisfies the imposed constraints). If no un-routed customer can be inserted into the current route, a new route is created. The process of creating new routes is repeated until all customers are routed.
3. New harmony improvisation: To create a new harmony, an empty solution X is created, and a random number *r* between zero and one is generated. If *r* is less than HMCR, a route is randomly selected from the best solution stored in HM and added it to X. Otherwise, a route is randomly created and appended to X. Any route selected from the best solution in HM is improved with regard to the PAR as follows:
   First, a random number *r* between zero and one is generated. If *r* is less than PAR, a neighborhood operator is randomly selected and applied to the current route while respecting VRPTW constraints. Since improvising a new solution in the SGHSA is based on the best solution in HM, if the size of the created solution X in terms of the number of routes equals to the best solution in HM, the improvisation process will be terminated. The created solution X is usually infeasible, because VRPWT is a constrained problem and given the process of improvising a new solution X, routes of a new solution X are either routes of the best solution in HM or randomly generated. So, with a very high probability, there are

customers who are either duplicated or deleted on these routes. A repair mechanism is employed to convert this infeasible solution into a feasible solution by eliminating the repeated customers and the assignment of the lost customers while maintaining VRPTW constraints. Figure 2 shows an example of the process of repair mechanism for 9 nodes. In this example, the node 1 represents depot and nodes 2 to 9 represent customers.

4. The harmony memory is updated. The quality of the solution X is calculated and replaced with the worst solution in HM and record the values of HMCR and PAR, if X has better quality than another one.
5. Termination criterion is checked. Steps 3-5 are repeated iteratively until the stopping condition based on the learning period is reached. Then, the best-solution obtained will be returned.



$Solution_1$
$Solution_2$
$Solution_{HMS}$
HM
Best solution : route 1: 1 5 6 2 1; route 2: 1 3 7 1; route 3: 1 8 4 9 1
New solution : route 1 : ; route 2 : ; route 3 :
Route 1 with probability 1-HMCR is randomly created and appended to new solution
Route 2 with probability HMCR is randomly selected from the best solution stored in HM and added it to new solution
Route 3 with probability 1-HMCR is randomly created and appended to new solution
New solution : route 1 : 1 5 6 8 1; route 2 : 1 8 4 9 1; route 3 : 1 3 7 5 1
New solution is infeasible, because Customers 5 and 8 are duplicate and customer 2 is missed.
repair mechanism : delete the duplicated customers from last routes and re-insert the missing customers into any feasible route and if they cannot be inserted to any of the already existing routes, create a new route for them.
infeasible solution : route 1 : 1 5 6 8 1; route 2 : 1 4 9 2 1; route 3 : 1 3 7 1

Figure 2. Repair mechanism

3.2. Local search algorithms

Local search algorithms (LS) have an important role in the exploitation of the search space used to intensify the search process in order to improve the algorithm convergence. In this work, these algorithms are adopted to improve SGHSA exploitation and applied to the improvised solution. For this purpose, the improvised solution is used as an initial solution for the LS algorithms. Unlike other methods, LS algorithms utilize several neighborhood structures at each iteration to generate a new neighbor solution. This procedure is iteratively repeated until the LS stopping condition is achieved. The LS neighborhood structures are (Bräysy & Gendreau, 2005a):

Relocate: Transfers a customer from one route to another.

Exchange: Swap two customers in different routes.

End Customer Interchange: Swap two end customers in different routes.

Or-opt: Relocate a chain of L consecutive customers so that three customers in the original tour are replaced by new three ones.

2-opt: Exchanges the end segments of two routes so that the last customers of a route are introduced after the first customers of another route.

CROSS-exchange: First, remove customers *i-1*, *i*, *k* and *k+1* from a first route while customers j-1, j, l and l+1 are removed from a second route. Then, the segments i–k and j–l, which may contain an arbitrary number of customers are swapped, are swapped as is shown in Figure 3.

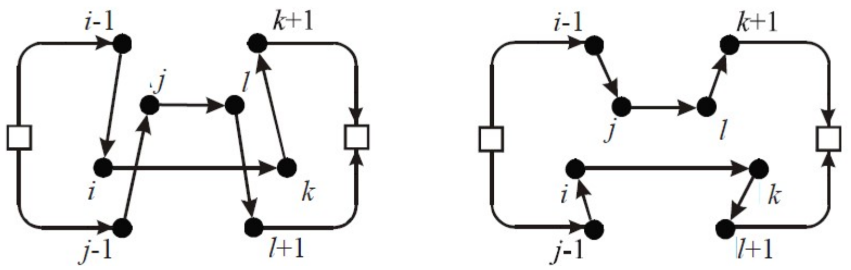


Figure 3. CROSS-exchange

The probability of P is used to apply LS algorithm, which is calculated by the following equation:

$$P = \frac{iteration}{maximum\ iteration} \qquad (1)$$

3.2.1 SA

SA (Kirkpatrick et. al., 1983) is a randomized method that has major advantage of its ability to avoid falling into local minima in comparison to other methods used in this paper. According to the annealing acceptance criterion, SA probabilistically accepts worse solutions in order to flee from the local optima. This algorithm begins with an initial solution X and iteratively generates a neighborhood solution X' via six neighborhood structures. If the quality of X' is better than X, this solution is replaced with X'. Otherwise, the worse solution might be accepted based on a certain probability *p* calculated as follows:

$$P = e^{\frac{\Delta f}{t}} \qquad (2)$$

Where Δf is the difference between the quality of X and X', and *t* is the current temperature. At each iteration, the temperature is decreased using the geometric scheduled as follows:

$$t = t \times \alpha \qquad (3)$$

Where α is the cooling rate ($\alpha < 1$). The search process is repeated until the stopping criterion is met (see Figure 4 (Yassen et. al., 2015a)).

```
Simulated annealing algorithm
1.  Input: Cooling schedule $\alpha$ ;
2.  $X = X_0$ ; /* Given of the initial solution $X_0$ */
3.  $T = T_{max}$; /* Starting temperature */
4.  while Stopping criteria is not met do
5.          Generate a neighborhood solution X';
6.          $\Delta f = f(X') - f(X)$
7.          If $\Delta f < 0$
8.                  X=X' ; /*Accept the neighbor solution*/ ;
9.          else
10.                 Accept X' with a probability $e^{-\frac{\Delta f}{t}}$
11.         end
12.         $T = T \times \alpha$ /* Temperature update*/;
13. end
14. Output Best solution found ;
```

Figure. 4. The pseudo-code of SA

3.2.2. GD

GD (Dueck, 1993) is a generic algorithm applied to optimization problems which is similar in many ways to the hill-climbing and simulated annealing algorithms. The main difference GD with the SA is the deterministic acceptance function of neighboring solutions. At each iteration, a generated neighbor solution is accepted if its quality is less than the current boundary value, named level (see Figure 5) (Talbi, 2009). The initial objective function is considered as the initial value of the level. The value of the level is monotonically decreased during the search using rain speed (UP) via Equation 4.

$$level = level - UP \quad (4)$$

Great deluge algorithm

```
1.  Input:
2.  $X = X_0$ ; /* Given of the initial solution $X_0$ */
3.  Choose the rain speed UP ; /* UP > 0 */
4.  Choose the initial water level LEVEL ;
5.  while Stopping criteria is not met do
6.          Generate a neighborhood solution X';
7.          If              LEVEL
8.                X=X' ; /*Accept the neighbor solution*/ ;
9.          end
10.         LEVEL = LEVEL − UP ; /* update the water level */;
11. end
12. Output: Best solution found.
```

Figure 5. The pseudo-code of GD

3.2.3. HC

HC (Simion, 2013) is a simple optimization algorithm which starts with an initial solution X and iteratively create a neighbor solution X' using the six neighborhood structures. If the quality of X' is better than X, substitute X with X'. Otherwise, X' is discarded and a new iteration is started (see Figure 6) (Yassen et. al., 2015a; Gehring & Homberger, 2001). The search process will be repeated until the stopping criterion is met.

Hill climbing algorithm

```
1. $X = X_0$ ; /* Given of the initial solution $X_0$ */
2. while Stopping criteria is not met do
3.       Generate N(X) /* Generation of candidate neighbors */ ;
4.       if there is no better neighbor then
5.             Stop
6.       end
7.       X=X' /*Select a better neighbor X' of  N(X)*/;
8. end
9. Output Final solution found (local optima).;
```

Figure 6. The pseudo-code of HC

3.3. HSGHSA

HSGHSA is a hybrid evolutionary algorithm whose performance is dependent on the efficiency of three major components including, HSA parameters, the type of LS algorithm and the LS neighborhood structures. These components have outstanding contribution in creating balance between exploitation and exploration and increase the ability of HSGHSA to solve various optimization problems. The process of adjusting these components is very difficult depending on the nature of the problem that will be discussed and no specific values of these components, which work well on all samples, exist. To ensure proper selection of HSA parameters in the first component, we use SGHSA that employs an adaptive parameter tuning method and parameters with regard to the VRPTW problem which are self-adapted through a learning mechanism or dynamic. Type of LS algorithm in the second component is guaranteed to select simulated annealing, hill climbing and great deluge LS algorithms that had good performance on the VRPTW (Yassen et. al., 2015a, 2015b, 2015c) (Bent & Van Hentenryck, 2004) (Baños et. al., 2013; Ding et. al., 2012; Clarke et. al., 1964; Osman, 1993) and is randomly determined to hybridize with SGHSA. Since LS plays such an important role in the algorithm's performance, we use several neighborhood structures. The HSGHSA method presented in Figure 7 is described in the following steps:

1. Initialize the parameter values of SGHSA (*HMS, LP, NI*, $HMCR_m$, PAR_m, $HMCR_{std}$ and PAR_{std}), and LS algorithms.
2. Harmony memory initialization and evaluation.
3. New harmony improvisation.
4. Apply local search: To improve the improvised solution X, an LS algorithm is used. The LS algorithm generates a neighbor's solution, using six neighborhood structures, that is respectively applied to the improvised solution X. The neighbor's solution will replace X if the quality of neighbor's solution is better than that of X; otherwise, the neighbor's solution will be rejected.
5. The harmony memory is updated. The quality of the solution X is calculated and replaced with the worst solution in HM and record the values of HMCR and PAR, if X has better quality than the bad one.
6. Termination criterion is checked. Steps 3-6 are repeated iteratively until the stopping condition based on the learning period is reached. Then, the best obtained solution will be returned.

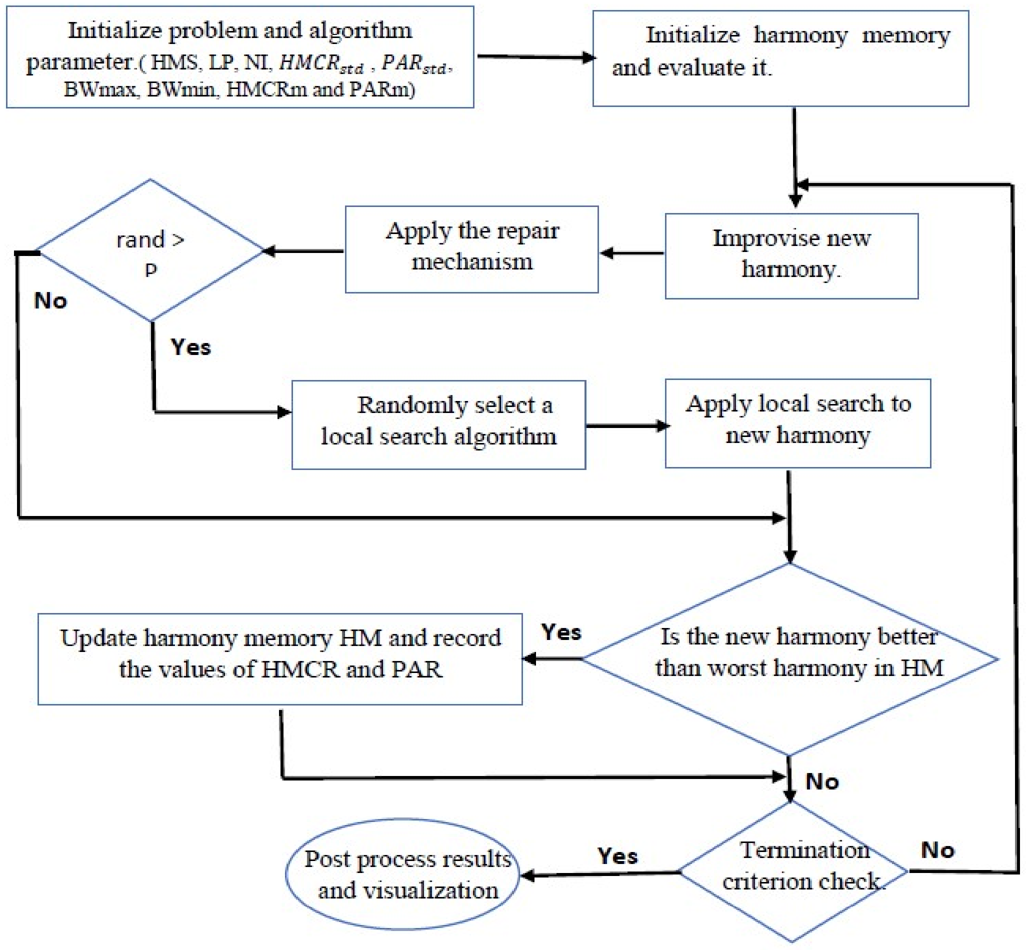

Initialize problem and algorithm parameter.(HMS, LP, NI, $HMCR_{std}$, PAR_{std}, BWmax, BWmin, HMCRm and PARm)
Initialize harmony memory and evaluate it.
Improvise new harmony.
Apply the repair mechanism
rand > p
No
Yes
Randomly select a local search algorithm
Apply local search to new harmony
Update harmony memory HM and record the values of HMCR and PAR
Yes
Is the new harmony better than worst harmony in HM
No
Post process results and visualization
Yes
Termination criterion check.
No

Figure 7. The flow chart of HSGHSA

4. Experimental design

We used Solomon's VRPTW benchmark (Solomon, 1987) to evaluate the performance of the HSGHSA. This benchmark contains 56 instances of Solomon with 100 customers, which are divided on six groups depending on the geographic location of the customers (R1, R2, C1, C2, RC1, RC2). The instances belonging to R1 and R2 groups located in random positions and C1 and C2 groups are in clusters. The RC1 and RC2 groups contain a mix of both random and clustered customers. The features of these instance sets are illustrated in Table 2.

Table 2. The characteristics of Solomon's VRPTW datasets

	Dataset					
Features	**R1**	**R2**	**C1**	**C2**	**RC1**	**RC2**
Number of Instants	12	11	9	8	8	8
Number of Customers	100	100	100	100	100	100
Number of vehicle	25	25	25	25	25	25
Capacity of Vehicle	200	1000	200	700	200	1000
Distribution of Customers	Random	Random	Cluster	Cluster	Random/ Cluster	Random/ Cluster

Since the process of setting parameters plays an important role in achieving optimal solutions, in this section the parameter setting of the proposed algorithm is considered. The parameter values of HSGHSA are self-adaptive and are adjusted during the evolution, but the initial values of parameters SGHSA and LS algorithms should be adjusted. These parameters include *HMS, LP, NI,* $HMCR_m$, PAR_m, $HMCR_{std}$ and PAR_{std} for SGHSA and α, t, UP for LS algorithms. The proposed HSGHSA is coded in Matlab 2014 and run on the 2.40 GHz Intel core i3 processor with Windows 8 operating system. The initial values of parameters $HMCR_m$, PAR_m, $HMCR_{std}$ and PAR_{std} are 0.98, 0.9,0.01 and 0.05 respectively, as suggested in (Quan-Ke Pan et. al., 2010). Although these values in reference (Quan-Ke Pan et. al., 2010) are for continuous optimization problems, but due to the self-compatibility of these parameters in the SGHSA, these values are updated during the run in accordance with the problem conditions.

The values of the parameters HMS and LP are fixed in 20 and 100 based on the preliminary test, respectively (see Table 3). This test is performed on six varied instances (R1-01, R2-01, C1-09, C2-06, RC1-01 and RC2-01) with 10 times execution of the proposed algorithm.

Table 3: The results of the different HMS and LP values

HMS	LP	R1-01	R2-01	C1-09	C2-06	RC1-01	RC2-01	Average
5	50	1650.7	1207.5	856.22	638.53	1705.7	1370.9	1238.3
	100	1646.5	1184.3	862.23	642.25	1699.9	1370.0	1234.2
	200	1649.6	1211.6	862.56	653.37	1714.3	1369.3	1243.5
10	50	1636.5	1203.1	874.67	650.95	1699.3	1351.8	1236.1
	100	1649.8	1201.1	837.39	694.21	1686.2	1364.8	1238.9
	200	1651	1207.9	886.96	645.95	1683.7	1351.4	1237.8
20	50	1641.8	1186.4	891.06	632.99	1697.3	1336.0	1230.9
	100	1663.8	1188.7	838.45	686.09	1673.6	1332.8	**1230.6**
	200	1655.5	1205.2	866.03	672.80	1689.3	1335.2	1237.3

GD has only one parameter that is the rain speed parameter (UP) calculated by Equation 5 as suggested in (Yassen et. al., 2015a). SA algorithm has two parameters, initial temperature (t) and the cooling schedule (α) fixed to 100 and 0.9, respectively.

$$UP = \frac{f(\textbf{initial solution}) - \textbf{estimated}(\textbf{lower bound})}{number\ of\ iteration} \qquad (5)$$

5. Experimental Results

In this section, we will investigate the effectiveness of the HSGHSA in tackling VRPTW. For this purpose, In Section 5.1 the results obtained by the HSGHSA have been tabulated and compared with the best-known results in the literature and state-of the-art methods. In Section 5.2, the results of the HSGHSA are compared to the standard HSA and its variants. Finally, in Section 5.3, we compare the results of the HSGHSA with HSA-HC, HSA-SA, HSA-RTS and Meta-HSA (Yassen et. al., 2015a, 2015b, 2015c).

The stopping criterion of the proposed method is decided based on the running time as suggested in (Yassen et. al., 2015a, 2015b, 2015c), which has been fixed in 25 minutes. The stopping criterion for the LS algorithms is adopted according to the number of non-improved iterations (NI_L=1000). Because the proposed approach is a metaheuristic algorithm, the results are reported for 31 independent runs to obtain an accurate statistical analysis and the best solution found for all instances is reported.

5.1. Evaluation of HSGHSA performance and comparison with other methods

Tables 4 show the results obtained by the proposed HSGHSA method for instances belonging to each data set C1, R1, RC1, C2, R2, and RC2, respectively. The first column of the table is the instances name, and column 2 is the value of the best-known solution (BKS) in the literature and is divided into 3 sub-columns a, b and c. Column a, b and c give information about total distance traveled (TD), number of vehicles (NV) and the reference (Ref) of the heuristic. Column 3 gives information about the results obtained by HSGHSA which is divided into 5 sub-columns a, b, c, d and e. Column a, b and c give information about best, average (Ave) and standard deviation (Std) in 31 runs respectively. Column *d* gives the number of vehicles *NV*, column *e* gives the percent deviation (Gap) between the best obtained solution and best-known. The Gap is calculated as follows:

$$Gap = \left(\frac{v1 - v2}{v2}\right) \times 100 \qquad (6)$$

Where $v1$ is the best result obtained by HSGHSA and $v2$ is the best-known value (TD) in the literature. A negative value of Gap indicates that the best result obtained by the proposed method is better than the best-known on an instance and a positive value of Gap indicates that the best-known is better than the best result obtained by proposed method. Finally, the zero represents the same performance methods.

According to the results listed in Table 4 and percentage deviation, we can see that HSGHSA has attained 5 best-known solutions shown in bold and has improved R1-01 best-known solution with percentage deviations -1.05. Furthermore, the proposed algorithm was able to obtain very close solutions to BKS in other instances such that Gap of 18 instances is less than 4 percentages.

Table 4: The results obtained by the HSGHSA on Solomon's VRPTW benchmark

		BKS				HSGHSA				
Data set		a. TD	b. NV	c. Ref		a. Best	b. Avg	c. Std	d. NV	e. Gap%
C1-01		828.94	10	2*		**828.94**	843.11	25.03	10	0
C1-02		828.94	10	2*		**828.94**	896.56	31.41	10	0
C1-03		828.06	10	2*		857.63	919.34	29.04	10	3.57
C1-04		824.78	10	2*		888.5	949.78	29.72	10	7.72
C1-05		828.94	10	2*		**828.94**	831.05	4.66	10	0
C1-06		828.94	10	2*		830.33	845.27	25.80	10	0.16
C1-07		828.94	10	2*		**828.94**	834.23	11.87	10	0
C1-08		828.94	10	2*		831.93	896.32	38.88	10	0.36
C1-09		828.94	10	2*		831.83	881.87	27.13	10	0.34
R1-01		1642.87	20	5*		**1625.6**	1662	12.623	20	-1.05
R1-02		1472.62	18	5*		1480.3	1523.2	14.32	18	0.52
R1-03		1213.62	14	2*		1241.2	1271.8	14.35	15	2.27
R1-04		982.01	10	2*		1046.6	1083.4	18.64	11	6.57
R1-05		1360.83	15	5*		1390.8	1430.2	16.27	16	2.20
R1-06		1241.518	13	5*		1281	1307.3	12.44	15	3.18
R1-07		1076.125	11	5*		1133.3	1161.1	17.37	11	5.31
R1-08		948.573	10	5*		988.49	1017.7	14.63	11	4.20
R1-09		1151.839	13	5*		1217.2	1268.7	24.74	13	5.67

R1-10		1080.36	11	2*		1131	1177.1	19.59	12	4.68
R1-11		1053.496	12	5*		1101.2	1160.3	19.91	12	4.52
R1-12		953.63	10	2*		1029.8	1059.1	16.96	11	7.98
RC1-01		1623.58	15	2*		1670.6	1704.2	22.83	16	2.89
RC1-02		1466.84	14	5*		1519.4	1547.9	16.13	15	3.58
RC1-03		1261.67	11	3*		1345.8	1412.8	24.82	12	6.66
RC1-04		1135.48	10	4*		1198.2	1248.7	17.48	11	5.52
RC1-05		1518.600	16	5*		1564.8	1606.4	20.18	16	3.04
RC1-06		1377.352	13	5*		1432.9	1452.2	19.01	14	4.03
RC1-07		1212.830	12	5*		1285.2	1321.4	17.04	13	5.96
RC1-08		1117.526	11	5*		1188.7	1229.7	22.12	11	6.36
C2-01		591.56	3	2*		**591.56**	655.64	29.15	3	0
C2-02		591.56	3	2*		609.94	653.96	25.04	3	3.10
C2-03		591.17	3	2*		620.81	700.43	25.98	3	5.01
C2-04		590.60	3	2*		632.89	700.74	43.52	3	7.16
C2-05		588.88	3	2*		591.42	644.48	27.72	3	0.43
C2-06		588.49	3	2*		615.74	674.53	28.97	3	4.63
C2-07		588.29	3	2*		598.77	639.33	26.913	3	1.78
C2-08		588.32	3	2*		619.24	682.22	37.66	3	5.25
R2-01		1147.80	9	1*		1184.3	1223.3	22.41	9	3.18
R2-02		1039.32	5	1*		1071.5	1188.4	245.19	6	3.09
R2-03		874.87	5	1*		915.64	961.57	57.121	6	4.66
R2-04		735.8	3	1*		814.21	952.54	240.66	5	10.65
R2-05		954.160	5	1*		1014.6	1053.6	20.48	6	6.33
R2-06		884.25	4	1*		945.75	1008.7	49.21	4	6.95
R2-07		797.99	4	1*		861.62	948.33	144.63	6	7.97
R2-08		705.62	3	1*		772.67	856.03	217.96	4	9.50
R2-09		860.11	5	5*		912.19	994.38	214.25	6	6.05
R2-10		910.98	5	1*		959.99	1020.2	31.38	6	5.37
R2-11		755.82	4	1*		823.73	885.91	37.31	4	8.98
RC2-01		1266.11	9	1*		1302.2	1346	21.23	8	2.85
RC2-02		1096.75	8	1*		1130.8	1200.4	28.711	7	3.10
RC2-03		926.89	5	1*		996.87	1061.1	113.9	6	7.55
RC2-04		786.38	4	1*		849.94	988.70	207.29	4	8.08
RC2-05		1157.55	7	1*		1240.8	1296.7	34.25	8	7.19
RC2-06		1056.21	7	1*		1124.3	1173.1	34.54	7	6.44
RC2-07		966.08	7	1*		1060.6	1143.2	77.87	6	9.78
RC2-08		779.84	4	5*		876.05	949.19	48.89	5	12.33

1* - De Olivera, Vasconcelos, Alvarenga, Mesquita, & De Souza, 2007
2* - Rochat & Taillard, 1995
3* - Shaw, 1998
4* - Cordeau, Laporte, & Mercier, 2001
5* - Alvarenga, Mateus, & Tomi, 2007

In addition, in order to demonstrate the efficiency of the algorithm, six of the solutions found in the examples in the previous tables are presented in Figure 8. This figure shows the results of HSGHSA for the R101, R201, C101, C201, RC101, and RC201 instances in which NV is the number of vehicles and TD is the total distance. It should be noted that in the R1-01 examples presented in this figure, the proposed algorithm has been able to improve the optimum solution compared to the best-known solution.

Furthermore, the HSGHSA results are compared to those obtained by other state-of-the-art methods in the literature. We select twelve works that have obtained the best results on VRPTW in Table 5.

These methods contain the following:

- **Meta-HSA**: Hybrid harmony search algorithm (Yassen et. al., 2015a)
- **NB10:** Penalty-based edge assembly memetic algorithm (Nagata et. al., 2010)
- **Homberger:** Hybrid ant colony with tabu search (Homberger & Gehring, 2002)
- **BVH:** Two-stage hybrid local search (Bent & Van Hentenryck, 2004)
- **CH01: P**arallel two-phase metaheuristic (Gehring & Homberger, 2001)
- **CH99**: Two evolutionary meta-heuristics (Homberger & Gehring, 1999)
- **CLM: U**nified tabu search (Cordeau et. al, 2001)
- **B01:** Reactive variable neighborhood search (Bräysy, 2003)
- **EA2:** Fast evolutionary metaheuristic (Bräysy & Dullaert, 2003)
- **LC03: C**ooperative parallel meta-heuristic (Le Bouthillier & Crainic, 2005)
- **LCK05: G**uided cooperative search (Le Bouthillier et. al., 2005)
- **PDR09: B**ranch-and-price-based large neighborhood search algorithm (Prescott-Gagnon et. al., 2009)

Table 5 presents the best solutions obtained via the application of meta-HSA, Homberger, CLM, CH01, CH99, B01, BVH, EA2, LC03, LCK05, PDR09, NB10 and HSGHSA on Solomon's benchmark datasets including, R1, R2, C1, C2, RC1 and RC2 alongside the published best results. Each row in this table contains 3 parts of the average total distances obtained for each dataset (TD), the percentage of deviation from the published best (%TD) and the average number of vehicles for each dataset(NV). The method for calculating percentages is given below.

$$\%TD = \left(\frac{TD_h - TD_b}{TD_b}\right)\times 100 \quad (7)$$

where TD_h is distance of heuristic concerned and TD_b is distance of best solution.

The results show that the proposed HSGHSA has been able to find better solutions than other algorithms for two group instances, R2 and RC2 out of six. Furthermore, in 4 groups, including R1, C1, C2 and RC1, although the algorithm cannot obtain better solutions than others, the high quality solutions are gained and the quality of the solutions are acceptable compared with most of the algorithms.

In more details, the proposed algorithm has obtained the solutions with 3.44, 6.31, 1.34, 3.42, 4.59 and 6.79 presents of %TD respectively in comparison with the published best solutions. Although HSGHSA did not manage to beat the best-known results for all datasets, the obtained results for these datasets are very competitive.

Table 5: Comparison of the performance of the proposed HSGHSA and different heuristics

Algorithms		**R1**	**R2**	**C1**	**C2**	**RC1**	**RC2**
Published Best	NV	13.08	4.73	10.00	3	12.75	6.38
	TD	1181.45	878.79	828.38	589.86	1339.24	1004.48
Meta-HSA	TD	1207.76	977.19	838.47	605.41	1381.96	1099.12
	%TD	+2.23	+11.19	+1.22	+2.64	+3.19	+9.42
	NV	-	-	-	-	-	-
Homberger	TD	1226.38	969.96	828.38	589.86	1392.57	1144.43
	%TD	+3.80	+10.3	0	0	+3.98	+13.93
	NV	12.00	2.73	10.00	3.00	11.63	3.25
CLM	TD	1210.14	969.57	828.38	589.86	1389.78	1134.52
	%TD	+2.43	+10.33	0	0	+3.77	+12.95
	NV	12.08	2.73	10.00	3.00	11.50	3.25
CH01	TD	1217.57	961.59	828.63	590.33	1395.13	1139.73

	%TD	+3.06	+9.42	+0.03	+0.08	+4.17	+13.46
	NV	12.00	2.73	10.00	3.00	11.50	3.25
CH99	TD	1198	947	829	590	1365	1144
	%TD	+1.40	+7.76	+0.07	+0.02	+1.92	+13.89
	NV	1242	2.82	10.00	3.00	11.88	3.25
B01	TD	1222.12	975.12	828.38	589.86	1389.58	1128.39
	%TD	+3.44	+10.96	0	0	+3.6	+12.34
	NV	11.92	2.73	10.0	3.00	11.50	3.25
BVH	TD	1231.08	954.18	828.38	589.86	1384.17	1124.47
	%TD	+4.20	+8.58	0	0	+3.35	+11.95
	NV	12.18	2.73	10.00	3.00	11.50	3.25
EA2	TD	1220.14	977.57	828.38	589.86	1397.44	1140.06
	%TD	+3.27	+11.24	0	0	+4.35	+13.49
	NV	12.00	2.73	10.00	3.00	11.50	3.25
LC03	TD	1209.19	963.62	828.38	589.86	1389.22	1143.70
	%TD	+2.35	+9.65	0	0	+3.73	+13.86
	NV	12.08	2.73	10.00	3.00	11.50	3.25
LCK05	TD	1214.20	954.32	828.38	589.86	1389.22	1143.70
	%TD	+2.77	+8.59	0	0	+3.73	+13.86
	NV	11.92	2.73	10.00	3.00	11.50	3.25
PDR09	TD	1210.34	955.74	828.38	589.86	1384.16	1119.44
	%TD	+2.45	+8.76	0	0	+3.35	+11.44
	NV	11.92	2.73	10.00	3.00	11.50	3.25
NB10	TD	1210.34	952.08	828.38	589.86	1384.72	1119.45
	%TD	+2.44	+8.34	0	0	+3.39	+11.45
	NV	11.92	2.73	10.00	3.00	11.50	3.25
HSGHSA	TD	1222.20	934.2	839.55	610.04	1400.7	1072.69
	%TD	+3.44	+6.31	+1.35	+3.42	+4.59	+6.79
	NV	13.08	4.72	10.00	3.00	12.75	6.37

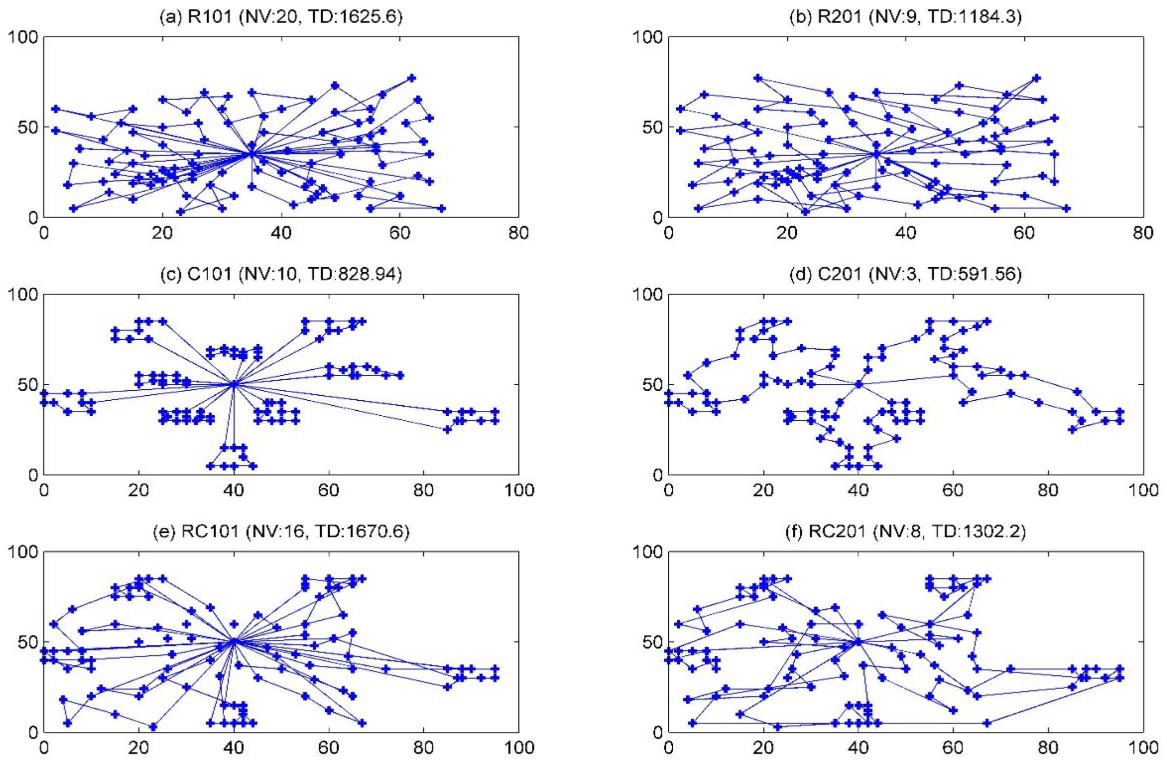


Figure 8. The results of instances: R101, R201, C101, C201, Rc101 and Rc201

The HSGHSA incorporates the local search algorithms in order to exploit local routing solutions. To demonstrate the effectiveness of local exploitation in HSGHSA, the convergence trace of the best and average the traveled distance in a population for six selected instances (one from each category) with and without the local search are plotted in Figure 9. As shown in Figure 9, the HSGHSA hybrid with local search performs better by having lowered the traveled distance for almost all instances than the one without any local exploitation. It has also been observed that other instances in the Solomon's 56 data sets exhibit similar convergence performances as those shown in Figure 8, which confirm the importance of incorporating local search exploitation in HSGHSA.

5.2. Comparisons of the HSGHSA with the standard HSA its variants

According to various methods of parameter setting, many versions of HSA were presented in the literature. In this subsection, we investigate the performance of the HSGHSA and compare it with the standard HSA and seven other well-known HSA variants including Classic HAS, Improved harmony search algorithm (IHSA) (Mahdavi et. al., 2007), Global best harmony search algorithm (GHSA) (Omran & Mahdavi, 2008), SGHSA (Quan-Ke Pan et. al., 2010), Harmony search algorithm with dynamic subpopulation (DHSA) (Pan et. al., 2011), Modified harmony search algorithm (MHSA) (Cheng et. al., 2008), Self-adaptive harmony search algorithm (SHSA) (Yadav et. al., 2012), and Intelligent tuned harmony search algorithm (ITHSA) (Yadav et. al., 2012).

The results of the standard HSA and HSA variants have been collected from (Yassen et. al., 2015a). The standard HSA and the HSA variants use three LS neighborhood structures (Relocate, Exchange and Two-opt star). To show the method's performance more clearly, the Best, Avr and Std of the results of eight mentioned algorithms and the proposed algorithm are tabulated in Table 6 in which the best results of nine algorithms are highlighted in bold. Moreover, mean of Best, Ave and Std of all algorithms for nine instances are shown in this table.

The following remarks upon the computational of this table shows that the proposed method appears to be robust generating high-quality solutions for every benchmark instance. Precisely, the objective values of the HSGHSA method appear to be clearly lower than those obtained by the other algorithms. With the exception of the obtained results of benchmark instance by three algorithms, HSA, DHSA and MHSA, the proposed methodology produces higher quality solutions than other five algorithms. By According to the results listed, the performance of the HSGHSA in terms of all indexes, the best, average and standard deviation, is much better than the basic HSA and its variants. In fact, using six neighboring structures leads to enhance the exploitation ability instead of randomly using a neighborhood structure. Besides, the results of the columns HSGHSA and SGHSA show that even with the same parameter setting, the results are very different due to the use of different neighboring structures. It is noted that applying the self-adaptive method for parameters setting, have a significant impact on the obtained results.

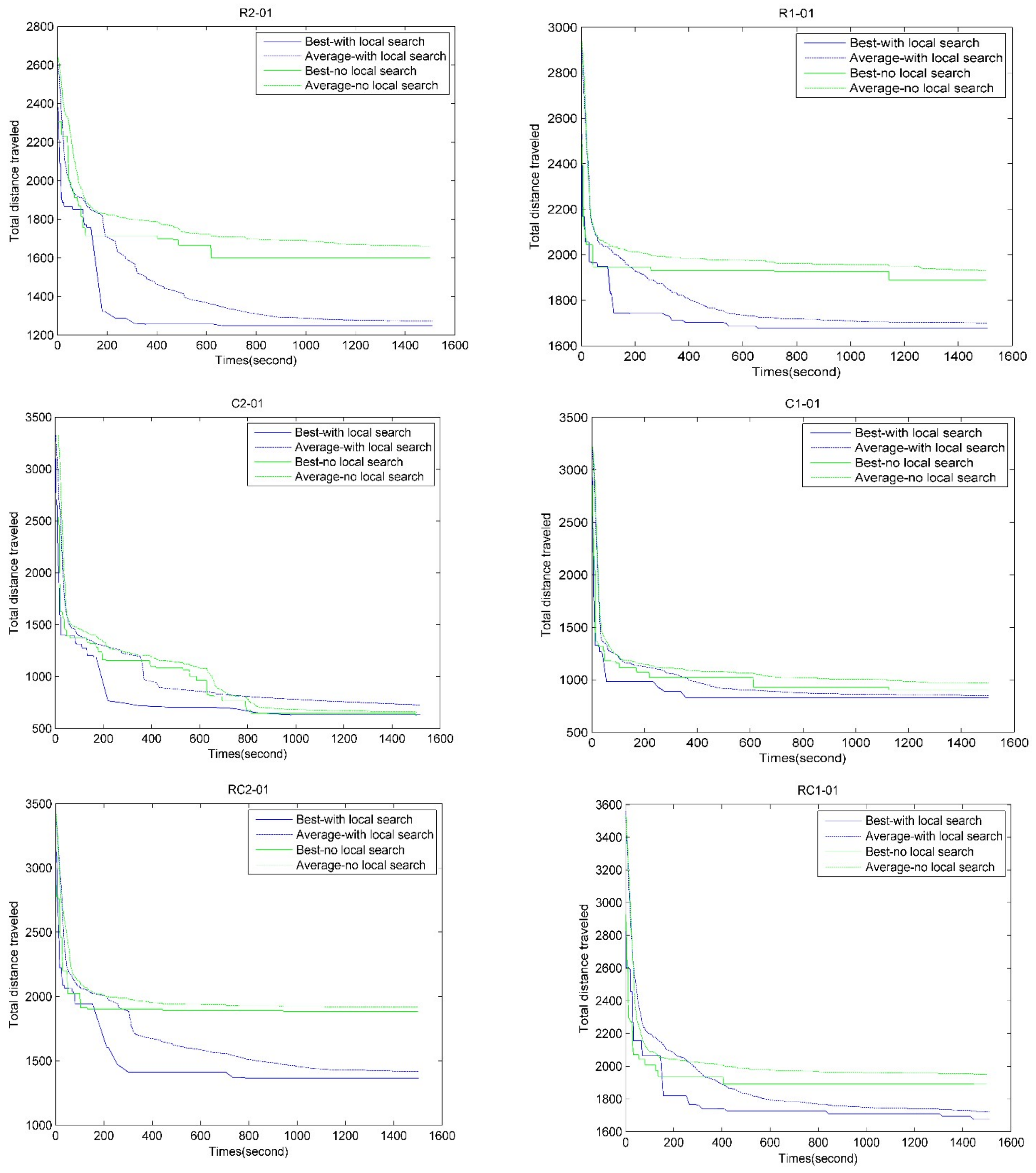


Figure 9. Comparison of simulations with and without local search exploitation in HSGHSA

5.3. Comparisons of HSGHSA with the proposed methods based on HSA to solve VRPTW in literature

In 2015, Yassen et al. proposed HSA-HC, HSA-SA, and HSA-RTS (Yassen et. al., 2015c) that hybridized HSA with LS algorithms and also Meta-HSA (Yassen et. al., 2015a) to solve VRPTW. In this section, the performance of HSGHSA is compared to these four hybrid schemes on eight different instances in Table 7. According to the results listed in this table, the tabulated comparison results show that all of the methods obtained the same results on instances R1-01, C1-02 and C2-01 with the exception of Meta-HSA for the benchmark instance C2-01 and HSGHSA for the benchmark instance R1-01. Besides, the results indicate that although both algorithms HSA-RTS and Meta-HSA can obtain three BKSs, the first one finds the best solutions for R1-01, C1-02 and C2-01 and the best solutions of R1-01, R2-01 and C1-02 are obtained by the second algorithm. Also, Meta-HSA outperformed HSA-HC, HSA-SA, HSA-RTS and HSGHSA on instance C2-06. Furthermore, HSA-SA have been able to find the BKSs in four examples out of eight, including R1-01, C1-02, C2-01 and RC2-01, but HSA-SA obtained the best results for the two instances, C1-09 and RC1-01. By comparing results of these algorithms, it is concluded that not only two best-known solutions on instances C1-02 and C2-01 are obtained, but also the algorithm can improve results of three instances, R1-01, R2-01 and RC2-01. Therefore, although the HSGHSA did not manage to beat the best-known results for all instances, the obtained results by the proposed algorithm for these instances are very competitive and generally better than HSA-HC, HSA-SA, HSA-RTS and Meta-HSA.

6. Conclusion

In this work, a hybridized SGHSA has been proposed to solve the VRPTW and is used to explore the search. Besides, the randomly LS algorithm is selected of three well-known LS algorithm and is used to further improve the solution generated by the SGHSA. The 56 VRPTW instances of Solomon's benchmark are used to evaluate the performance of the proposed method.

Table 6: The best, Ave and Std results of HSGHSA compared to the other algorithms

	HAS			**IHSA**			**GHSA**			**SGHSA**			**HSGHSA**		
Dataset	Best	Ave	Std	Best	Ave	Std	Best	Ave	Std	Best	Ave	Std	Best	Ave	Std
R1-01	1704.11	1762.02	29.13	1692.5	1767	37.05	2389.48	2534.3	85.81	2025.19	2245.04	104.94	**1625.6**	**1662**	**12.623**
R1-03	1387.59	1497.12	39.37	1412.28	1507.72	50.76	2125.59	2366.45	93.58	1646.71	1846.03	99.78	**1241.2**	**1271.8**	**14.358**
R2-01	1858.5	1990.66	75.73	1824.19	1983.83	96.69	2106.68	2294.26	85.78	1846.44	2150.17	133.94	**1184.3**	**1223.3**	**22.415**
C1-02	1228.2	1456.81	104	1176.51	1402.57	117.08	2708.38	2920.21	129.48	1616.49	2051.46	273.28	**828.94**	**896.56**	**31.415**
C1-09	1362.78	1615.31	133.59	1418.72	1659.52	111.6	2894.76	3079.97	111.26	1690.83	2127.45	248.93	**831.83**	**881.87**	**27.133**
C2-06	1516.25	1802.84	171.22	1505.35	1758.84	148.17	2109.02	2308.17	124.97	1677.9	2226.03	265.53	**615.74**	**674.53**	**28.971**
C2-08	1414.39	1679.24	130.67	1297.89	1666.59	153.07	1805.01	2161.45	130.52	1751.98	2091.27	242.35	**619.24**	**682.22**	**37.66**
RC1-01	1734.57	1814.3	48.28	1703.34	1811.43	42.16	2537.67	2742	101.4	2097..18	2325.94	135.17	**1670.6**	**1704.2**	**22.837**
RC2-01	2106.93	2237.13	89.22	2084.07	2260.14	90.68	2497.04	2701.32	111.01	2227.3	2448.81	158.03	**1302.2**	**1346**	**21.232**
Mean	1590.37	1761.71	91.25	1568.32	1757.52	94.14	2352.63	2567.57	108.20	1810.36	2168.02	184.66	**1102.18**	**1149.16**	**24.30**

	DHSA			**MHSA**			**SHSA**			**ITHSA**			**HSGHSA**		
Dataset	Best	Ave	Std	Best	Ave	Std	Best	Ave	Std	Best	Ave	Std	Best	Ave	Std
R1-01	1704.11	1956.56	57.73	1692.5	1836.07	43.84	2389.48	1741.15	21.65	2025.19	1771.57	31.61	**1625.6**	**1662**	**12.623**
R1-03	1387.59	1613.44	57.72	1412.28	1586.46	44.7	2125.59	1447.27	23.83	1646.71	1485.82	34.75	**1241.2**	**1271.8**	**14.358**
R2-01	1858.5	2024.42	140.67	1824.19	2099.55	95.18	2106.68	1521.87	37.16	1846.44	1991.73	130.52	**1184.3**	**1223.3**	**22.415**
C1-02	1228.2	1623.65	147.43	1176.51	1614.25	147.89	2708.38	1197.21	55.87	1616.49	1334.46	121.16	**828.94**	**896.56**	**31.415**
C1-09	1362.78	1841.43	151.67	1418.72	1833.45	142.58	2894.76	1225.69	64.8	1690.83	1603.77	120.06	**831.83**	**881.87**	**27.133**
C2-06	1516.25	2051.09	199.4	1505.35	1968.11	213.28	2109.02	888.88	43.71	1677.9	2050.26	426.17	**615.74**	**674.53**	**28.971**
C2-08	1414.39	2021.68	171.32	1297.89	1938.39	166.3	1805.01	862.95	40.16	1751.98	2042.63	501.94	**619.24**	**682.22**	**37.66**
RC1-01	1734.57	1998.28	92.51	1703.34	1893.67	68.19	2537.67	1800.66	37.9	2097..18	1830.43	50.74	**1670.6**	**1704.2**	**22.837**
RC2-01	2106.93	2362.23	110	2084.07	2412.18	146.35	2497.04	1772.53	38.89	2227.3	2282.6	105.77	**1302.2**	**1346**	**21.232**
Mean	1590.37	1943.64	125.38	1568.32	1909.3	118.70	2352.63	1384.25	40.44	1810.36	1821.48	169.19	**1102.18**	**1149.16**	**24.30**

Table 7: The best results of HSGHSA compared to the proposed methods based on HA.

Dataset	Best know	HSA-HC	HSA-SA	HSA-RTS	Meta-HSA	HSGHSA
R1-01	1642.88	**1642.88**	**1642.88**	**1642.88**	**1642.88**	***1625.6***
R2-01	1202.96	1203.61	1203.61	1258.61	**1202.96**	***1184.3***
C1-02	828.94	**828.94**	**828.94**	**828.94**	**828.94**	**828.94**
C1-09	828.94	831.79	**831.30**	866.40	832.29	831.83
C2-01	591.56	**591.56**	**591.56**	**591.56**	**-**	**591.56**
C2-06	588.49	644.32	650.44	699.22	594.7	619.24
RC1-01	1631.17	1632.20	**1631.17**	1660.22	1639.73	1670.6
RC2-01	1326.45	**1326.45**	1332.09	1366.48	1345.16	***1302.2***

The obtained results showed that the use of six neighborhood structures has a significant impact on results and the proposed method achieved competitive results compared to algorithms proposed in the literature and it even acts better than the other methods on R2 and RC2 dataset.

The other results also showed that the proposed method outperformed the standard HSA and its variants and it is an effective solution method for VRPTW. It seems that using effective heuristic algorithms can lead to gaining better solutions than other famous algorithm. Furthermore, this powerful metaheuristic can be used for solving for other versions of VRP. Future projects will focus on working on such ideas and making them operational.

References

Azi, N., Gendreau, M., Potvin, J. Y. (2010). *An exact algorithm for a vehicle routing problem with time windows and multiple use of vehicles*, European Journal of Operational Research 202, 756-763

Alvarenga, G. B., Mateus, G. R., Tomi, G. D. (2007). *A genetic and set partitioning two phase approach for the vehicle routing problem with time windows*, Computers & Operations Research 34, 1561–1584

Baños, R., Ortega, J., Gil, Fernández, A., & de Toro, F. (2013). *A Simulated Annealing-based parallel multi-objective approach to vehicle routing problems with time windows*, Expert Systems with Applications 40, 1696-1707

Bent, R., Van Hentenryck, P. (2004). *A two-stage hybrid local search for the vehicle routing problem with time windows*, Transp. Sci. 38 (4), 515-530

Blum, C., Puchinger, J., Raidl, G. R., & Roli, A. (2011). *Hybrid metaheuristics in combinatorial optimization: A survey*, Applied Soft Computing, 11(6):4135-4151

Bräysy, O., & Gendreau, M. (2005a). *Vehicle routing problem with time windows, part i: Route construction and local search algorithms*, Transportation science, 39 (1):104–118

Bräysy, O. (2003). *A reactive variable neighborhood search for the vehicle routing problem with time windows*, INFORMS J. Comput. 15, 347-368

Bräysy, O., & Dullaert, W. (2003). *A fast evolutionary metaheuristic for the vehicle routing problem with time windows*, Int. J. Artif. Intell. 12 (2), 153-172

Cheng, Y., Li, L., Lansivaara, T., Chi, S., & Sun, Y. (2008). *An improved harmony search minimization algorithm using different slip surface generation methods for slope stability analysis*, Engineering Optimization, 40(2):95–115

Clarke, G., & Wright, J. W. (1964). *Scheduling of vehicles from a central depot to a number of delivery points*, Oper. Res. 12 (4), 568-581

Cordeau, J. F., Laporte, G., & Mercier, A. (2001). *A unified tabu search heuristic for vehicle routing problems with time windows*, J. Oper. Res. Soc. 52, 928-936

Dariusz, B. (2014). *A cooperative population learning algorithm for vehicle routing problem with time windows.* Neurocomputing 146, 210-229

De Olivera, H. C. B., Vasconcelos, G. C., Alvarenga, G. B., Mesquita, R. V., & De Souza, M. M. (2007). in: J.-F. Puget (Ed.), Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Scheduling (CI-Sched), Springer-Verlag, New York, 417-431

Ding, Q., Hu, X., Sun, L., & Wang, Y. (2012). *An improved ant colony optimization and its application to vehicle routing problem with time windows,* Neurocomputing 98, 101-107

Dueck, G. (1993). *New optimization heuristics: The great deluge algorithm and the record-torecord travel*, Journal of Computational Physics, 104(1):86-92

Fu, L.L., Aloulou, M.A. & Triki, C. (2017). *Integrated production scheduling and vehicle routing problem with job splitting and delivery time windows*, International Journal of Production Research, 1-16.

Gao, K.Z., Suganthan, P.N., Pan, Q.K. and Tasgetiren, M.F. (2015). *An effective discrete harmony search algorithm for flexible job shop scheduling problem with fuzzy processing time*, International Journal of Production Research, 53(19), 5896-5911

Geem, Z. W., Kim, J. H., & Loganathan, G. (2001). *A new heuristic optimization algorithm: harmony search*, Simulation, 76(2):60–68

Gehring, H., Homberger, J. (2001) *A parallel two-phase metaheuristic for routing problems with time windows*, Asia-Pac. J. Oper. Res. 18, 35-47

Homberger, J., & Gehring, H. (1999). *Two evolutionary meta-heuristics for the vehicle routing problem with time windows*, Inf. Syst. Oper. Res. 37, 297–318

Homberger, J., & Gehring, H. (2002). *Parallelization of two phase metaheuristic for routing problems with time windows*, Journal of Heuristics 8, 251-276

Kirkpatrick, S., Gelatt, C. D., Vecchi, M. P., et al. (1983). *Optimization by simmulated annealing*, science, 220(4598):671-680

Le Bouthillier, A., & Crainic, T. G. (2005). *A cooperative parallel meta-heuristic for the vehicle routing problem with time windows*, Comput. Oper. Res. 32 (7), 1685-1708

Le Bouthillier, A., Crainic, T. G., & Kropf, P. (2005). *A guided cooperative search for the vehicle routing problem with time windows*, IEEE Intell. Syst. 20 (4), 36-42

Lin, J., Liu, M., Hao, J. & Gu, P. (2017). *Many-objective harmony search for integrated order planning in steelmaking-continuous casting-hot rolling production of multi-plants*, International Journal of Production Research, 55(14), 4003-4020

Mahdavi, M., Fesanghary, M., & Damangir, E. (2007). *An improved harmony search algorithm for solving optimization problems*. Applied mathematics and computation, 188(2):1567–1579

Nagata, Y., Bräysy, O., & Dullaert, W. (2010). *A penalty-based edge assembly memetic algorithm for the vehicle routing problem with time windows*, Computers & Operations Research 37, 724- 737

Nagata, Y. (2007). *Edge assembly crossover for the capacitated vehicle routing problem*, Proceedings of the 7th European conference on evolutionary computation in combinatorial optimization, 124-53

Nagata, Y, & Bräysy, O. (2009). *A powerful route minimization heuristic for the vehicle routing problem with time windows*, Operations Research Letters, in press

Nagata, Y. (2006). *New EAX crossover for large TSP instances*, Proceedings of the 9th international conference on parallel problem solving from nature, 372–81

Omran, M. G. and Mahdavi, M. (2008). *Global-best harmony search*, Applied Mathematics and Computation, 198(2):643–656

Osman, I. H. (1993). *Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem*, Ann. Oper. Res. 41, 421-451

Pan, Q. K., Suganthan, P. N., Tasgetiren, M. F., & Liang, J. J. (2010). *A self-adaptive global best harmony search algorithm for continuous optimization problems*, Applied Mathematics and Computation 216, 830-848

Pan, Q. K., Suganthan, P. N., Liang, J. J., & Tasgetiren, M. F. (2011). *A local-best harmony search algorithm with dynamic sub-harmony memories for lot-streaming flow shop scheduling problem*, Expert Systems with Applications, 38 (4): 3252–3259

Prescott-Gagnon, E., Desaulniers, G., & Rousseau, L. M. (2009). *A branch-and-price-based large neighborhood search algorithm for the vehicle routing problem with time windows*, Networks 54 (4), 190-204

Rochat, Y., & Taillard, ´E. D. (1995). *Probabilistic diversification and intensification in local search for vehicle routing*. Journal of heuristics,1(1): 147–167

Shaw, P. (1998). *Using constraint programming and local search methods to solve vehicle routing problems*, in: M. Maher, J.-F. Puget (Eds.), Principles and Practice of Constraint Programming, Lecture Notes in Computer Science

Simon, D. (2013). Chapter 2 of *Evolutionary Optimization Algorithms Biologically-Inspired and Population-Based Approaches to Computer Intelligence*, Published by John Wiley & Sons, Inc., Hoboken, New Jersey.

Solomon, M. M. (1987). *Algorithms for the vehicle routing and scheduling problems with time window constraints*. Operations research, 35 (2):254-265.

Talbi, E. G. (2009). From design to implementation: Wiley Online Library

Ursani, Z., Essam, D., Cornforth, D., & Stocker, R. (2011). *Localized genetic algorithm for vehicle routing problem with time windows*, Applied Soft Computing 11, 5375–5390

Wang, C. M. & Huang, Y.F. (2010). *Self-adaptive harmony search algorithm for optimization*, Expert Systems with Applications, 37(4):2826– 2837

Yassen, E. T., Ayob, M., Nazri, M. Z. A., Sabar, N. R. (2015a). *Meta-Harmony search algorithm for the vehicle routing problem with time windows*, Information Sciences, DOI: 10.1016/j.ins.2015.07.009

Yassen, E. T., Ayob, M., & Nazri, M. Z. A. (2015b). *A Hybrid Meta-Heuristic Algorithm for Vehicle Routing Problem with Time Windows*, International Journal on Artificial Intelligence Tools Vol. 24, Issue 6, 1550021

Yassen, E. T., Ayob, M., & Nazri, M. Z. A. (2015c). *The effect of hybridizing local search algorithms with time windows*, Journal of Theoretical and Applied Information Technology

Yadav, P., Kumar, R., Panda, S. K., & Chang, C. (2012). *An intelligent tuned harmony search algorithm for optimization*, Information Sciences, 196:47–72

Yu, B., Yang, Z. Z., Yao, B. Z. (2011). *A hybrid algorithm for vehicle routing problem with time windows*, Expert Systems with Applications 38, 435-441