Iraqi Journal for Computer Science and Mathematics

Journal Homepage: http://journal.esj.edu.iq/index.php/IJCM
e-ISSN: 2788-7421 p-ISSN: 2958-0544

Tackling the Berth Allocation Problem via Harmony Search Algorithm

Bilal Ahmed[1], Hazlina Hamdan[1]*, Abdullah Muhammed[2], Nor Azura Husin[1]

[1]Department of Computer Science, Faculty of Computer Science and Information Technology,
Universiti Putra Malaysia, 43400 UPM Serdang, MALAYSIA
[2]Department of Communication Technology and Networks, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, 43400 UPM Serdang, MALAYSIA

*Corresponding Author: Hazlina Hamdan

DOI: https://doi.org/10.52866/ijcsm.2024.05.03.031
Received March 2024; Accepted May 2024; Available online August 2024

ABSTRACT: Berth Allocation Problem (BAP) is a renowned difficult combinatorial optimization problem that plays a crucial role in maritime transportation systems. BAP is categorized as non-deterministic polynomial-time hard (NP-hard) problems, that is very tough to resolve for optimality within an acceptable timeframe. Many metaheuristic algorithms have been suggested to tackle this problem, and yet, most of these algorithms have some drawbacks such as they have a weak ability to explore the solution space (they struggle escaping from local minima) and they face the difficulties to operate on different datasets. Consequently, the need to either enhance the existing algorithms or utilize a new algorithm is still necessary. Harmony Search Algorithm (HSA) is one of the recent population-based optimization methods which inspired by modern-nature. HSA has confirmed its ability to tackle various difficult combinatorial optimization problems like vehicle routing, exam timetabling to name a few. However, as far as we are concerned, it has never been applied to tackle the BAP problem. The primary objective of this article is to examine the effectiveness of HSA in solving BAP by identifying suitable values for the parameters of the HSA and then applying HSA to tackle BAP. Therefore, in this article, the basic HSA is proposed to tackle the BAP. The suggested HSA is tested on BAP benchmark (I3 dataset) and compared the results with other latest algorithms found in the literature. The trial outcomes evidenced that the HSA is promising, competitive, and that it has surpassed some other algorithms that have solved the same dataset, and the results were very near to the best-known results. Experimental results also prove the suitability and applicability of HSA in tackling the BAP.

Keywords: Berth allocation problem, combinatorial optimization problems, harmony search algorithm, metaheuristic algorithms, modern nature inspired algorithm.

1. INTRODUCTION

Berth scheduling constitutes a vital aspect of daily operations within container terminals [1]. The Berth Allocation Problem (BAP) emerges as a challenging combinatorial optimization problem within maritime transportation systems, playing a pivotal influence on vessel handling efficiency. Effective BAP resolution is imperative, as bad choices can lead to unwarranted delays and carrier dissatisfaction, considering berths' significant role as a critical resource in maritime transportation [2]. The BAP involves the task of allocating suitable berthing location and handling schedule for a specified number of ships, adhering to all BAP constraints while optimizing designated objective functions [3]. It is all about where and when the vessels should be anchored (see Figure 1). The most often utilized objective functions of BAP are minimizing operational costs, increasing seaport productivity, and minimizing the aggregate service and waiting times for all vessels [4, 5].

The BAP is a non-deterministic polynomial-time hard (NP-hard) optimization problem that is hard to resolve optimally in an acceptable timeframe [6, 7]. For NP-hard tasks, the complexity of search time grows proportionally with problem size [8]. Consequently, scholars tend for using metaheuristic algorithms to solve BAP as they can attain significant success in getting high-quality solutions, in a reasonable time frame [9].

A relatively recent metaheuristic algorithm, HSA, has got a growing attention amid numerous scholars as a cutting-edge population-based algorithm that inspired by modern-nature to address diverse hard combinatorial optimization

*Corresponding author: hazlina@upm.edu.my problems [10]. Imitating the way musicians improvise their musical tones, HSA iteratively seeks a state of harmony or pleasance (optimal solution) aligned with artistic criteria [11].

The motives that stimulate us to use HSA are as following [12, 13]: (i) HSA has successfully tackled an enormous variety of optimization problems across diverse fields and has demonstrated its capacity to handle challenging combinatorial optimization problems. (ii) HSA is considered as simple, flexible, and it requires less mathematical operations. (iii) It requires less computational resources, particularly memory and execution time. (iv) Most importantly, HSA offers a good-balanced strategy for enhancing both local and global exploration capabilities via its improvisation operators.

In this article, an investigation into the application of HSA to tackle the BAP is underway. Based on the information available and in accordance with the latest scientific literature, HSA has never been used to address the BAP until the time of writing. Accordingly, this stimulates us to explore the performance of HSA for solving BAP. Hence, it is worthy to assess the HSA for this task. The suggested HSA is assessed with a benchmark dataset called I3 dataset that were used by other scholars and verified by comparing it against latest algorithms reported in the literature to confirm its performance.

The article is arranged as follows:- Problem description of BAP is explained in Section two, whereas the suggested harmony search algorithm to tackle the BAP is shown in Section three. Computational findings and discussion are given in Section four. Last but not least, Section five shows the conclusion and future works.

2. PROBLEM DESCRIPTION

The Berth Allocation Problem (BAP) stands as a globally recognized NP-hard combinatorial optimization problem encountered by all seaports worldwide [9]. Regarded as a pivotal optimization issue, the BAP can be categorized based on spatial/temporal considerations of berths and vessels [5]. Concerning spatial constrains, there are two types: discrete and continuous berth models. In the discrete berth model, the quayside is segmented to a limited number of docking spaces, with each berth accommodating a single vessel, and strict one-to-one allocation is observed. Conversely, the continuous berth model considers the whole quay itself as a single berth (there is no portioning along the quay) allowing vessels to moor at any position within the quay boundaries wherever empty spaces permit, as shown in Figures 2 and 3. Concerning temporal constrains, there are two types: static and dynamic berth model. In the static BAP, all vessels are presumed to have arrived at the harbor and be ready for service prior to plan the berth assignment, obviating the need for specified arrival times. Conversely, in the dynamic BAP, the timing of vessel arrivals is uncertain but their scheduled arrival times are predictable. Therefore, the vessels cannot berth before the expected arrival times [14]. The primary objective in solving the BAP is to allocate all vessels to available berths with minimal waiting time, while adhering to the following constraints [6, 15, 16]:

- Every single berth can only serve a single ship at one time.
- Any ship with a certain handling time can be allocated to any berth, taking into consideration that the ship's handling time varies from one berth to another.
- All ships with known arrival time can reach to the berth at any time regardless of its opening hours.
- When a ship is anchored at a berth, it will stay at the berth depending on when the service handling is completed.

The objective function of the BAP is calculated as indicated in Equation (1) [14]:

$$\sum_{i \in N} \sum_{k \in K} (\boldsymbol{t}_i^k - \boldsymbol{a}_i + \boldsymbol{P}_i^k \sum_{j \in N \cup \{d\}} \boldsymbol{x}_{jj}^k) \tag{1}$$

Where:

- a_i : *the arrival time of vessel i*
- K : *set of berths,* $K = \{1,2,\ldots.,K\}$
- *N: number of vessels that will arrive at the harbor,*
- P_i^k: *the handling time of vessel i at berth k*
- T_i^k: *the berthing time of vessel i at berth k*
- x_{ij}^k :*decision variable,*x_{ij}^k*=1 if vessel j is served by berth k immediately after vessel i*

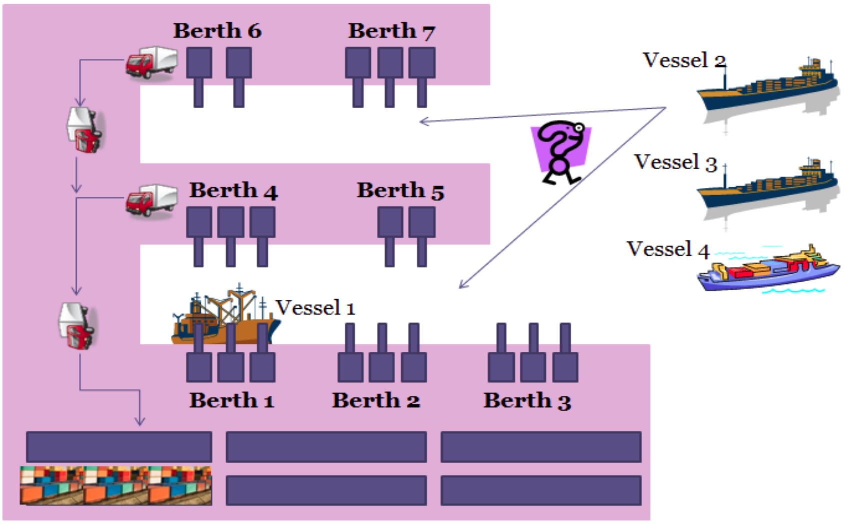


FIGURE 1. **Berth allocation problem schematic of a container terminal**

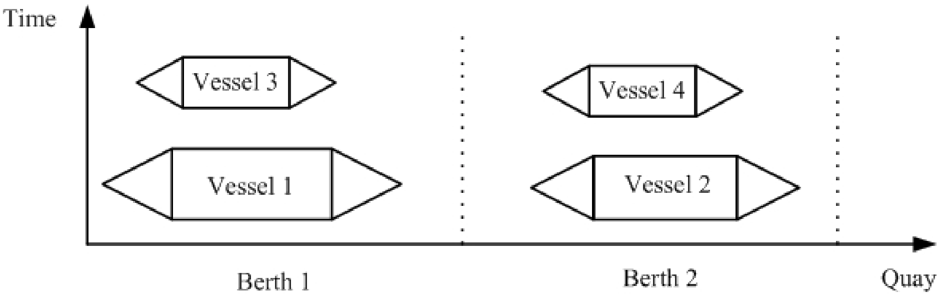


FIGURE 2. **Discrete berth allocation problem**

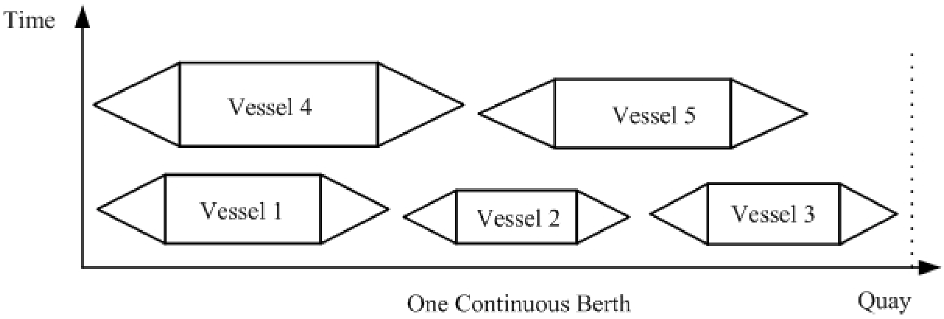


FIGURE 3. **Continuous berth allocation problem**.

3. HARMONY SEARCH ALGORITHM FOR THE BERTH ALLOCATION PROBLEM

The Harmony Search Algorithm (HSA), which introduced by Zong Woo Geem and Joon Woong Kim, stands out as a relatively recent metaheuristic approach [17]. HSA is novel optimization algorithm that inspired by modern-nature. HSA draws inspiration from the artistic practice of musical improvisation progression where musicians adjust their notes to achieve a pleasing harmony, to seek an optimal solution via recurrent trials, achieving the condition of harmony/pleasance aligned with artistic criteria. This parallels the optimization process's pursuit of an optimal solution, guided by an objective function. Within the music progress, a player selects and combines a diverse set of pitches based on prior knowledge, subsequently rendering these through musical instruments to attain an optimum harmonious musical expression. This procedure parallels an optimal design process for addressing optimization problems, wherein harmony is parallel to the optimization solution, and the improvisations resemble local and global search process [18].

Like a bunch of musicians refining their harmonies through successive practices, the HSA enhances solution vectors iteratively by leveraging discovered candidate solutions during the solution construction phase. Iteratively, HSA creates a fresh harmony (solution) for a certain problem, mirroring the incremental development of musical harmonies in practice sessions. In the realm of musical improvisation, each player faces three potential choices when performing a composition [19]: (i) performing a well-known musical composition from memory, (ii) playing notes closely resembling a familiar composition (with slight pitch adjustments), or (iii) creating notes entirely at random. Analogously, HSA iteratively creates new solutions by following three guiding principles: (i) memory consideration, (ii) pitch adjustment, and (iii) random consideration. The parallelism between musical improvisation and optimization problems is depicted in Figure 4. The fundamental structure of HSA consists of 6 steps as shown in Figure 5.

- Step #1: Initializing the parameters of HSA.
- Step #2: Initializing the Harmony Memory (HM).
- Step #3: Improvising a new solution (harmony).
- Step #4: Updating Harmony Memory (HM).
- Step #5: Verifying the termination criteria.
- Step #6: Cadenza.

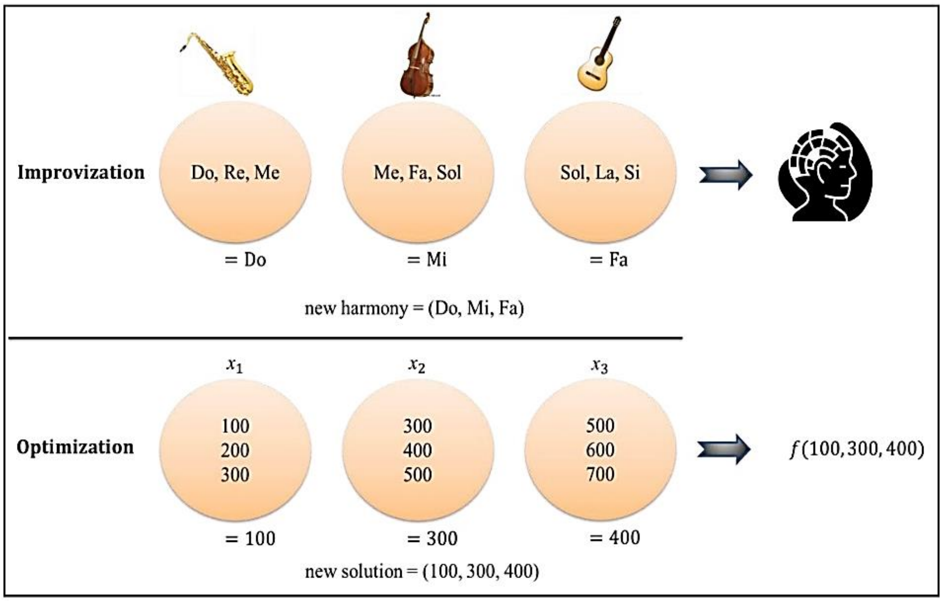


FIGURE 4. **Flowchart of the components of the basic HSA**

Step #1: *Initializing the parameters of HSA*

In this stage, the Harmony Search Algorithm's (HSA) parameters are set. is governed by four key parameters [10, 13]:

- Harmony Memory Size (HMS). This parameter determines the maximum capacity for storing potential solutions (harmonies) in the Harmony Memory (HM). It resembles the population size in Evolutionary Algorithms (EAs).
- Harmony Memory Consideration Rate (HMCR). Employed in the improvisation process (solution generation), HMCR guides the selection of the decision variable that will be used to generate the next solution. It determines whether the variable will be chosen from the HM (with a likelihood of HMCR). The remaining probability (1-HMCR) allows for random selection of decision variables from the whole range of solution space. Usually HMCR falls within the range [0-1].

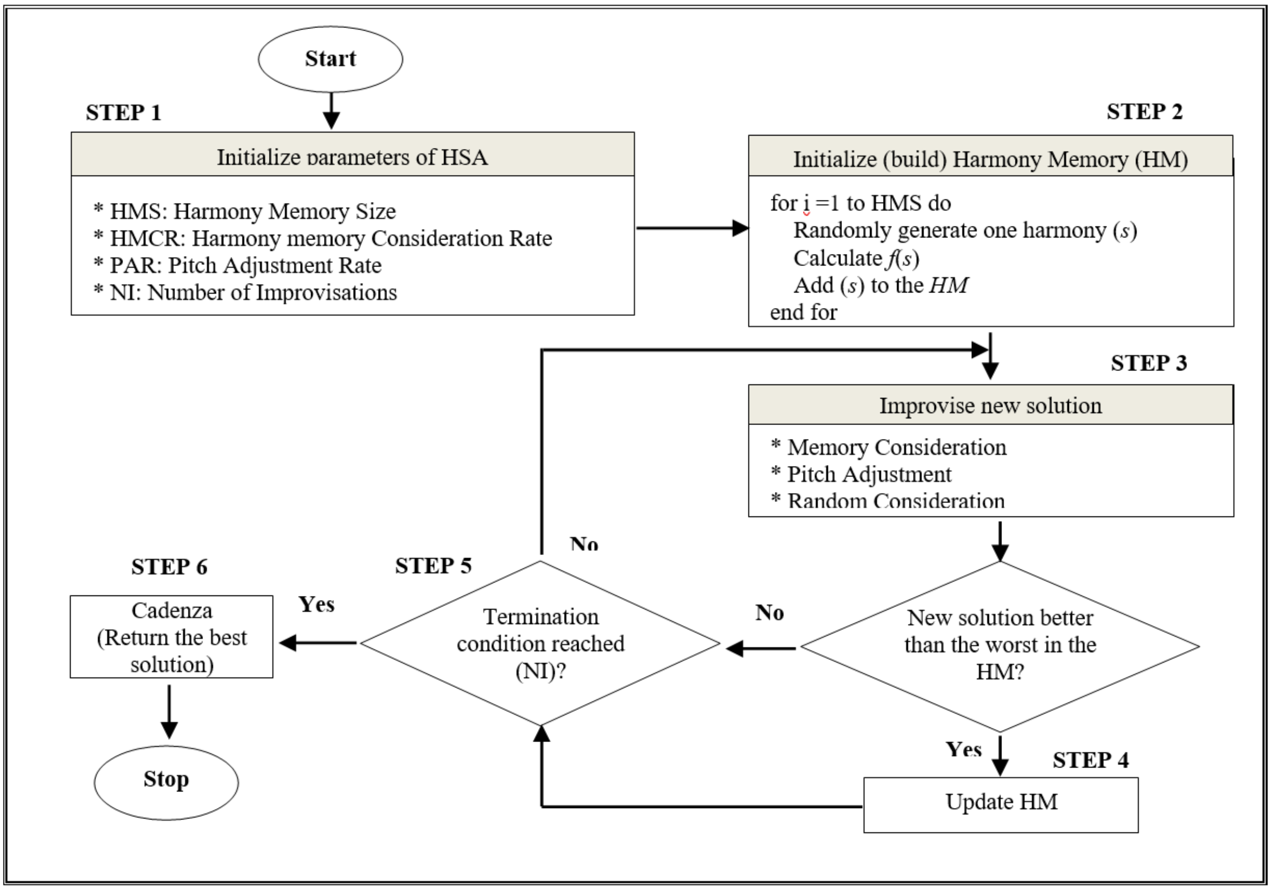

Start
STEP 1
Initialize parameters of HSA
* HMS: Harmony Memory Size
* HMCR: Harmony memory Consideration Rate
* PAR: Pitch Adjustment Rate
* NI: Number of Improvisations
STEP 2
Initialize (build) Harmony Memory (HM)
for i =1 to HMS do
Randomly generate one harmony (s)
Calculate f(s)
Add (s) to the HM
end for
STEP 3
Improvise new solution
* Memory Consideration
* Pitch Adjustment
* Random Consideration
New solution better than the worst in the HM?
Yes
STEP 4
Update HM
No
STEP 5
Termination condition reached (NI)?
No
Yes
STEP 6
Cadenza (Return the best solution)
Stop

FIGURE 5. **Flowchart of the components of the basic HSA**

- Pitch Adjustment Rate (PAR). This is also utilized in the improvisation process. When a random number is below the PAR threshold, the selected decision variable from the HM is modified to its neighboring value; else, it remains unchanged. PAR is set within the value [0, 1].
- Number of Improvisation (NI). This parameter specifies the maximum number of times the HSA algorithm will run, and it serves as a termination criterion.

Step #2: *Initializing the Harmony Memory (HM)*

Harmony memory symbolized as a two-dimensional matrix that stores a collection of solutions produced through a randomized process. Each row in the array embodies a distinct solution vector. The total number of solutions in the HM corresponds to HMS. The solutions are arranged in order of their objective function values, which can be either maximizing or minimizing, as illustrated in Equation 2.

$$HM = \begin{bmatrix} x_1^1 & x_2^1 & \dots & x_{N-1}^1 & x_N^1 \\ x_1^2 & x_1^2 & \dots & x_{N-1}^2 & x_N^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_1^{HMS-1} & x_2^{HMS-1} & \dots & x_{N-1}^{HMS-1} & x_N^{HMS-1} \\ x_1^{HMS} & x_2^{HMS} & \dots & x_{N-1}^{HMS} & x_N^{HMS} \end{bmatrix} \begin{bmatrix} f(x^1) \\ f(x^2) \\ \vdots \\ f(x^{HMS-1}) \\ f(x^{HMS}) \end{bmatrix} \tag{2}$$

Where x_1, x_2,…, x_{N-1}, x_N embody a solution vector, N is the number of possible solutions included in each solution vector, and $f(x^1)$, $f(x^2)$,…, $f(x^{HMS})$ denote the objective function values for each solution vector. The Harmony Memory (HM) stores solutions (harmonies) in descending order of their objective function values $f(X)$. In this article, the goal of the objective function is to reduce the overall amount of time ships spend at all berths in the port. The pseudocode of HM initialization is presented in Figure 6.

```
begin
  for i = (1 to HMS) do
    X_i = 0;
    for j = (1 to N) do     //N is the number of decision variables
      choose the value of $x_i^j$ randomly from the possible range
      add $x_i^j$ to $X_i$
    end for
    calculate the $f(x_i)$  //minimize total service time for all berths
    add $X_i$ to HM
  end for
  sort the solutions according to their objective function ascendingly
end
```

FIGURE 6. **Pseudocode for initializing the harmony memory**

Step #3: *Improvising a new solution (harmony)*

During this phase, a new harmony vector, x' = (x'_1, x'_2, x'_3,…, x'_N), which represents a potential solution, is improvised via a process called improvisation. This improvisation process follows three rules: (i) memory consideration, (ii) pitch adjustment, and (iii) random consideration. The selection mechanism used by HSA is depicted in Equation 3.

$$x_i^{New} \leftarrow \begin{cases} x_i(k) \in \{x_i(1), x_i(2), \dots, x_i(k_i)\} & \text{w.p } P_{\text{Random}} \\ x_i(k) \in \{x^1_i, x^2_i, \dots, x_i^{HMS}\} & \text{w.p } P_{\text{Memory}} \\ x_i(k \pm m) & \text{w.p } P_{\text{Pitch}} \end{cases} \tag{3}$$

Equation 3 outlines the selection mechanism employed by the HSA for generating new harmony vectors, which represent potential solutions. This mechanism incorporates three distinct considerations: random selection, memory-based selection, and pitch adjustment. As indicated in Equation 3, the value of variable i ($i = 1, 2, \dots, N$) is randomly selected from the available range $\{x_i(1), x_i(2), \dots, x_i(k_i)\}$ with a probability of P_{Random} (random consideration). Alternatively, it is chosen from the group of values stored in the Harmony Memory (HM) with a probability of of P_{Memory} (memory consideration). Additionally, there is a possibility of a slight adjustment by shifting to neighboring values $x_i(k \pm m)$, with a probability of P_{Pitch} (pitch adjustment) [10].

(i) Memory consideration

The creation of a new harmony, or solution, in the HSA is influenced by the value of the HMCR. A randomly generated number, R, within the range of [0 – 1] is generated. If R is less than the HMCR, the new solution is selected from the HM with a probability equal to the HMCR. This implies that the values of the decision variables (x'_1) are chosen from the corresponding values kept in the HM (x_1^1, x_1^2, x_1^3,…, x_1^{HMS}). On the other hand, if R is greater than or equal to the HMCR, the new harmony is randomly picked from the entire available range (X_i), as demonstrated in Equation 4.

$$x'_i \leftarrow \begin{cases} x'_i \in \{x_i^1, x_i^2, x_i^3, \dots, x_i^{HMS}\} & with\ probability\ HMCR \\ x'_i \in X_i & with\ probability\ 1 - HMCR \end{cases} \tag{4}$$

Where x'_i is a new value of x_i, and X_i is the whole available range.

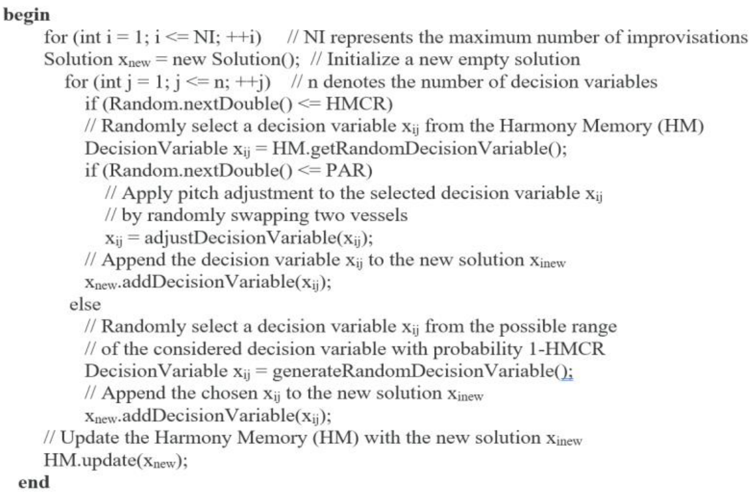
(ii) Pitch adjustment

Each decision variable (xi) selected using the HMCR is evaluated to determine whether it requires fine-tuning or pitch adjustment. With a probability of PAR (PAR ∈ [0, 1]), the modification is implemented by applying the move operator to modify xi to its neighboring value. In the absence of this adjustment, xi remains unchanged with a probability of (1-PAR). This process introduces a slight perturbation to the decision variables, preventing the algorithm stuck in local minima and promoting further exploration of the search space, as shown in Equation (5).

$$x'_i \leftarrow \begin{cases} x'_i + rand\ (0,1) * BW & with\ probability\ PAR \\ x'_i & with\ probability\ 1 - PAR \end{cases} \tag{5}$$

(iii) Random consideration

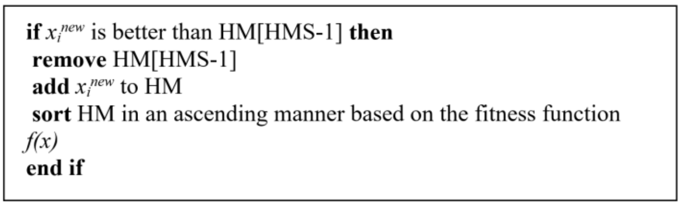
Solutions which are not chosen from the HM are arbitrarily chosen from the whole available solution space (with a probability of 1-HMCR), as illustrated in Equation 3. This randomness fosters a broader spectrum of possible solutions, allowing the algorithm to discover a wider range of the search space and potentially discover the global optimum. The pseudocode of the improvisation process is depicted in Figure 7.



```
begin
    for (int i = 1; i <= NI; ++i)     // NI represents the maximum number of improvisations
    Solution $x_{new}$ = new Solution();  // Initialize a new empty solution
      for (int j = 1; j <= n; ++j)    // n denotes the number of decision variables
        if (Random.nextDouble() <= HMCR)
        // Randomly select a decision variable $x_{ij}$ from the Harmony Memory (HM)
        DecisionVariable $x_{ij}$ = HM.getRandomDecisionVariable();
        if (Random.nextDouble() <= PAR)
          // Apply pitch adjustment to the selected decision variable $x_{ij}$
          // by randomly swapping two vessels
          $x_{ij}$ = adjustDecisionVariable($x_{ij}$);
        // Append the decision variable $x_{ij}$ to the new solution $x_{inew}$
        $x_{new}$.addDecisionVariable($x_{ij}$);
      else
        // Randomly select a decision variable $x_{ij}$ from the possible range
        // of the considered decision variable with probability 1-HMCR
        DecisionVariable $x_{ij}$ = generateRandomDecisionVariable();
        // Append the chosen $x_{ij}$ to the new solution $x_{inew}$
        $x_{new}$.addDecisionVariable($x_{ij}$);
    // Update the Harmony Memory (HM) with the new solution $x_{inew}$
    HM.update($x_{new}$);
  end
```

FIGURE 7. **Pseudocode of improvisation a new solution (harmony)**

Step #4: *Updating Harmony Memory (HM)*
The newly generated solution vector (x' = *x'*$_1$, *x'*$_2$, *x'*$_3$,.., *x'*$_N$).) is evaluated using the objective function *f(x)*. If the new solution surpassed the current solutions in terms of quality, it is added to the HM by substituting the worst solution (updating the HM). Subsequently, the HM is ordered to ensure that the new harmony occupies the appropriate location based on its quality. Conversely, if the newly generated solution fails to surpass existing solutions, it is ignored. Figure 8 shows the pseudocode of updating the HM.



```
if $x_i^{new}$ is better than HM[HMS-1] then
 remove HM[HMS-1]
 add $x_i^{new}$ to HM
 sort HM in an ascending manner based on the fitness function
f(x)
end if
```

FIGURE 8. **Pseudocode of updating the HM**

Step #5: *Verifying the termination criteria*
At this point, it is determined if the HSA has reached its stopping criterion. If the termination criterion has been met (i.e., the max number of iterations has been exceeded), HSA terminates. Otherwise, the updating and improvisation processes, as described in stage three and four, are reiterated till the termination condition is met.

Step #6: *Cadenza*
A cadenza is a virtuoso musical composition typically performed towards the end of a piece, showcasing the performer's improvisation skills and aiming to create the most pleasing harmonic effect. Similar to this musical concept, a cadenza in the HSA represents the final stage in the search process aiming to identify the top solution. During this stage, the HSA retrieves and yields the highest-quality solution discovered and kept in the harmony memory according to the objective function value *f(x)*. This solution represents the culmination of the algorithm's efforts to navigate the search space and uncover the most optimal solution.

4. EXPERIMENTAL RESULTS AND DISCUSSION

In order to investigate the standard HSA's efficacy in solving the BAP, the proposed HSA is evaluated using the BAP benchmark (I3 dataset) which contains 30 instances (represented as i1 to i30) derived from real-world data from the port of Gioia Tauro in Italy and generated randomly by Cordeau et al. in 2005 [15]. Each instance contains 60 vessels and 13 berths. The HSA was implemented using C# programming language performed on a laptop (Intel® Core i5 CPU @ 3.40GHz with 12 gigabyte RAM) operating under Windows 11 (64-bit) operating system. The efficacy of the HSA results is evaluated in comparison to recent algorithms obtained from earlier studies in the literature, i.e., Particle Swarm Optimization (PSO), Genetic Algorithm (GA) and Bird Mating Optimizer (BMO) [6, 14]. In this paper, the HSA was executed 35 independent runs for each instance. The rationale behind conducting the proposed algorithms 35 time is to secure a consistent measure of algorithm reliability and facilitate a sturdy statistical examination of algorithm performance [14, 20, 21].

Owing to the stochastic character of metaheuristic approaches, each problem domain necessitates an extra careful setting of the algorithm parameter values. Since there has been no investigation in the literature for the parameter value settings of HSA when it applies to solve BAP, a number of preliminary experimentations was run to find suitable parameter values that produce high-quality results. There are four parameters that HSA has: HMCR, HMS, PAR, and NI. According to the preliminary experiments carried out, the best parameter settings of the standard HSA for BAP (which are discovered after some preliminary experiments) are exhibited in Table 1.

Table 1. Parameter settings for the standard HSA (based on the best average results)

Parameter	Value
Harmony Memory Size (HMS)	40
Harmony Memory Consideration Rate (HMCR)	0.8
Pitch Adjustment Rate (PAR)	0.2
Number of Improvisations (NI)	800000

Table 2 illustrates the results obtained by the standard HSA tested on the I3 datasets. For each tested instance, the best solution (Best), average solution (Avrg), standard deviation (Stdv) and the relative percentage deviation (RPD) of the results generated by the HSA algorithm for all instances of the BAP are reported. The RPD is calculated as {(BOR - BKR)/BKR * 100%}, where BOR is the best obtained result from the comparing algorithms and BKR is the best results found in the scientific literature. Figure 9 draws the results showed in Table 2, which demonstrates the analysis of the range among BKR, Best and Avrg.

According to the results reported in Table 2, these results demonstrate that the standard HSA obtained promising results. It obtained competitive average results on most tested instances and produced a small standard deviation for nearly all BAP instances. This proves that HSA is a promising method for solving BAP. The box-whisker plot of solutions distribution for six instances of BAP (i01, i05, i10, i15, i20 and i25) over 35 runs is plotted to investigate the distribution of solutions obtained by the standard HSA, as described in Figure 10.

Table 2. Results obtained by the standard HSA for solving BAP

Instance	BKR	Best	Avrg	Stdv	RPD	Instance	BKR	Best	Avrg	Stdv	RPD
i01	1409	1428	1554.8	40.47	1.35%	**i16**	1364	1409	1545.7	42.56	3.30%
i02	1261	1263	1382.9	32.67	0.16%	**i17**	1283	1333	1465.1	44.41	3.90%
i03	1129	1144	1232.3	29.94	1.33%	**i18**	1345	1363	1489.1	38.83	1.34%
i04	1302	1333	1418.7	39.05	2.38%	**i19**	1367	1396	1530.5	41.21	2.12%
i05	1207	1219	1311.3	36.50	0.99%	**i20**	1328	1349	1488.7	39.12	1.58%
i06	1261	1276	1412.9	36.74	1.19%	**i21**	1341	1401	1502.9	35.58	4.47%
i07	1279	1298	1401.9	37.48	1.49%	**i22**	1326	1389	1514.8	42.36	4.75%
i08	1299	1343	1478.5	40.31	3.39%	**i23**	1266	1284	1417.7	35.23	1.42%
i09	1444	1465	1595.8	41.78	1.45%	**i24**	1260	1266	1392.5	34.27	0.48%
i10	1213	1225	1338.5	32.78	0.99%	**i25**	1376	1475	1638.4	40.85	7.19%
i11	1368	1419	1536.8	44.48	3.73%	**i26**	1318	1341	1477.3	37.01	1.75%
i12	1325	1356	1486.4	38.41	2.34%	**i27**	1261	1297	1397.4	39.32	2.85%
i13	1360	1365	1504.9	35.31	0.37%	**i28**	1359	1375	1512.5	36.67	1.18%
i14	1233	1248	1372.1	36.80	1.22%	**i29**	1280	1293	1413.1	36.54	1.02%
i15	1295	1305	1442.6	48.13	0.77%	**i30**	1344	1380	1524.1	36.20	2.68%

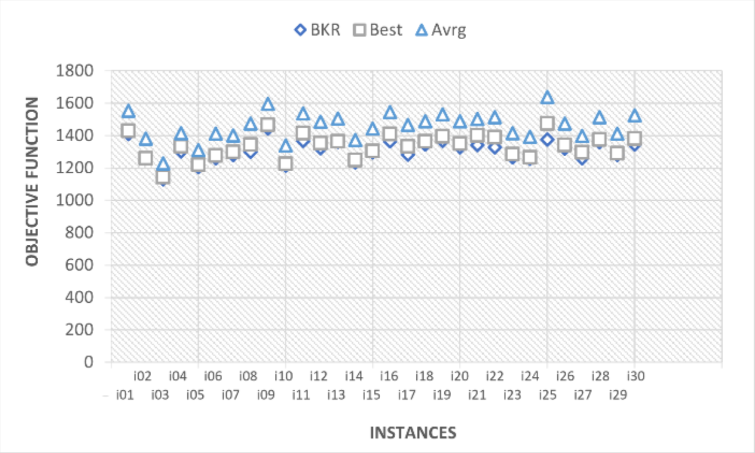


FIGURE 9. **The results of using the standard HSA for solving BAP**

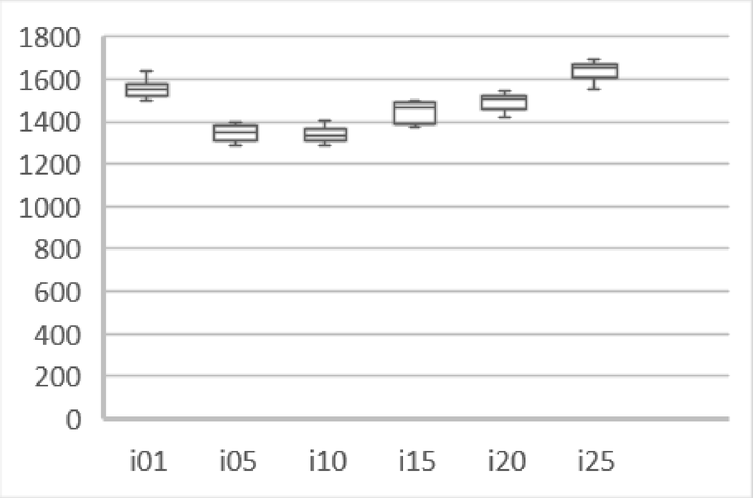


FIGURE 10. **The box-whisker plot of solution distribution of HSA for 6 instances of BAP**

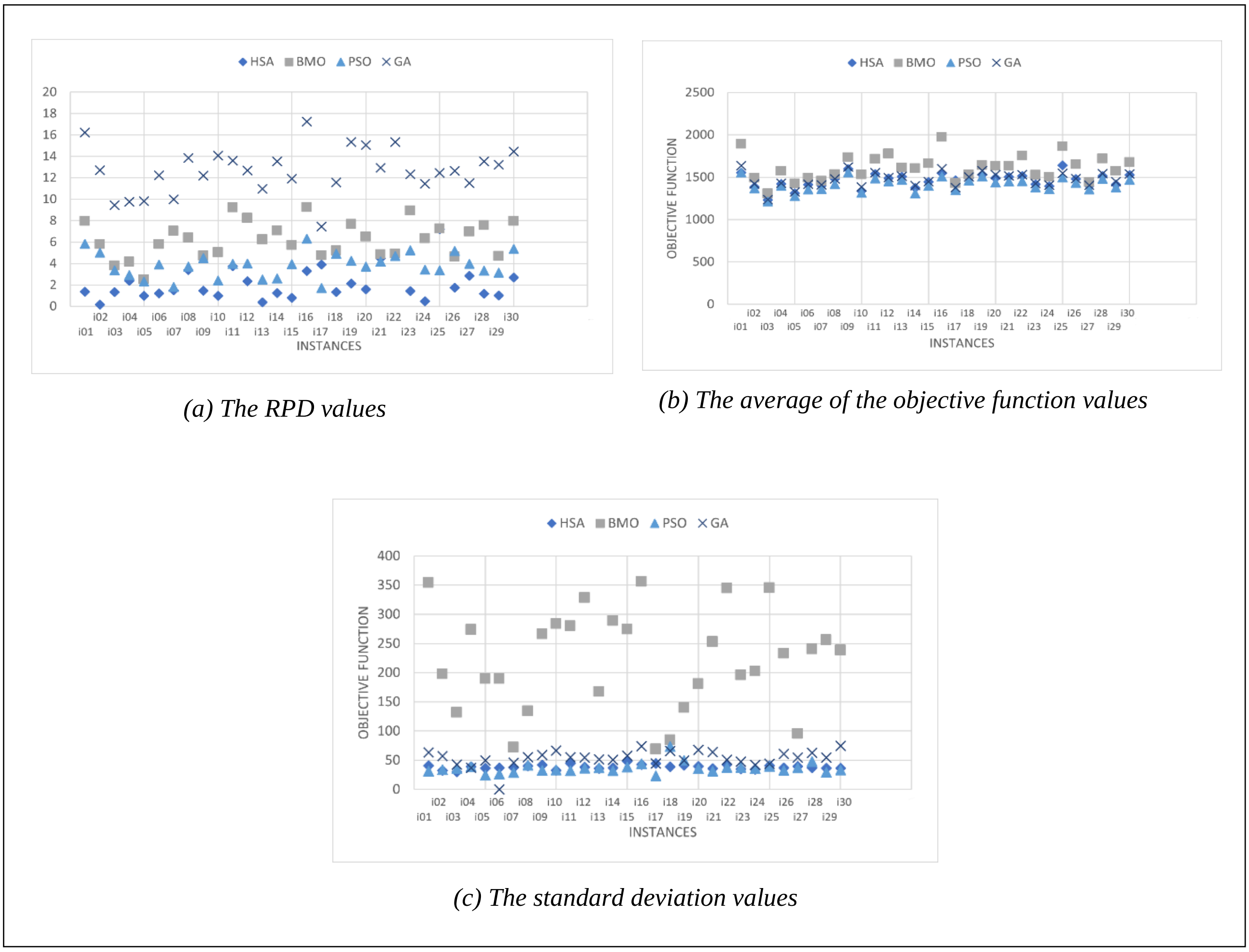
The plot shows that there are some instances in which the distribution of solutions is symmetric; The solutions exhibited a symmetrical distribution centered at the median (i.e., i05, i20). Regarding instances i01 and i10, the distribution of solution was skewed to the lower end (best obtained results). Finally, the solutions distribution of instances i15 and i25 was skewed to the upper end (worst solution). According to this plot, it is apparent that (for the majority of instances) HSA is a consistent algorithm in solving BAP. Table 3 exhibits a comparison between the results attained by the HSA against recent metaheuristic algorithms obtained from scientific literature. The highest outcomes are highlighted in boldface. Figure 11 (a), (b) and (c) show a comparison between HSA and other metaheuristics (in terms of RPD, Avrg and Stdv, respectively). Thus, there is an evidence that HSA has outperformed the BMO, PSO and GA in almost all instances in terms of the relative percentage deviation (26 instances out of 30). The average RPDs of the HSA, BMO, PSO and GA are 2.11, 6.24, 3.84 and 12.65, respectively which indicates that the results attained by HSA (over 35 executions) are comparatively not too far from the best-known results (BKR) (the BKR plus the BAP I3 dataset can be found in [7]).

Moreover, HSA surpassed the BMO and GA regarding the best results obtained and the average results throughout most instances (except instances i17 and i25 in GA). Pertaining to the standard deviation, HSA outperformed BMO and GA throughout all instances, and attained 11 instances out of 30, which is much better than the PSO. The average standard deviations of HSA, BMO, PSO and GA are 38.37, 222.18, 36.21 and 56.29, respectively, altogether signifying that the HSA is much more stable and consistent than BMO and GA (only PSO is better than HSA in terms of Stdv).

Table 3. Comparison results of HSA and other metaheuristics for solving I3 dataset

Instances	HSA			BMO			PSO			GA		
	Avrg	Stdv	RPD	Avrg	Stdv	RPD	Avrg	Stdv	RPD	Avrg	Stdv	RPD
i01	1554.8	40.47	1.35	1893	354.2	7.95	1552.8	31.35	5.82	1637.8	63	16.24
i02	1382.9	32.67	0.16	1491	197.91	5.79	1365.3	33.82	5	1421.4	57.2	12.72
i03	1232.3	29.94	1.33	1305	131.95	3.81	1211.6	35.46	3.37	1235.5	42.3	9.43
i04	1418.7	39.05	2.38	1572	273.71	4.15	1395.2	38.66	2.92	1429.1	36.9	9.76
i05	1311.3	36.50	0.99	1424	189.29	2.49	1273.5	24.32	2.32	1325.5	49.6	9.82
i06	1412.9	36.74	1.19	1492	189.23	5.79	1353.5	26.2	3.89	1415.4	43.1	12.24
i07	1401.9	37.48	1.49	1455	71.98	7.04	1356.7	28.68	1.8	1406.7	45.7	9.98
i08	1478.5	40.31	3.39	1533	134.01	6.39	1415.7	41.29	3.7	1478.9	55	13.85
i09	1595.8	41.78	1.45	1735	266.08	4.71	1554.7	32.34	4.5	1620.3	58.6	12.21
i10	1338.5	32.78	0.99	1531	283.86	5.03	1314.5	33.27	2.39	1383.8	66.6	14.08
i11	1536.8	44.48	3.73	1714	279.79	9.21	1481.4	31.69	3.95	1553.9	55.2	13.59
i12	1486.4	38.41	2.34	1777	328.52	8.23	1449.2	36.38	4	1493.1	54.6	12.69
i13	1504.9	35.31	0.37	1612	167.49	6.25	1466.9	36.94	2.5	1509.3	51.1	10.98
i14	1372.1	36.80	1.22	1604	288.49	7.06	1307.5	31.8	2.6	1399.8	50.7	13.53
i15	1442.6	48.13	0.77	1664	274.43	5.71	1395.7	37.86	3.94	1449.3	57.9	11.92
i16	1545.7	42.56	3.30	1974	355.87	9.24	1508.1	43.69	6.3	1599.4	73.9	17.26
i17	1465.1	44.41	3.90	1428	68.85	4.75	1343.7	23.09	1.71	1378.7	44.6	7.46
i18	1489.1	38.83	1.34	1528	84.37	5.2	1458.7	73.36	4.91	1500.7	65.5	11.58
i19	1530.5	41.21	2.12	1640	139.78	7.68	1507.4	49.35	4.24	1576.6	49.5	15.33
i20	1488.7	39.12	1.58	1631	180.85	6.48	1436.1	35.62	3.69	1528.1	67.5	15.07
i21	1502.9	35.58	4.47	1631	252.75	4.85	1448.3	30.93	4.18	1514.2	63.6	12.92
i22	1514.8	42.36	4.75	1755	344.73	4.9	1452.4	37.43	4.68	1529.5	50.7	15.35
i23	1417.7	35.23	1.42	1527	195.61	8.93	1378.7	37.76	5.21	1422.2	47.3	12.34
i24	1392.5	34.27	0.48	1501	202.77	6.35	1357.4	35.85	3.41	1404.2	41.7	11.44
i25	1638.4	40.85	7.19	1862	345.47	7.27	1492.9	39.64	3.34	1547.4	43.6	12.46
i26	1477.3	37.01	1.75	1650	233	4.63	1432.3	32.52	5.16	1484.7	61	12.65
i27	1397.4	39.32	2.85	1440	95.11	6.98	1352.2	36.99	3.97	1406.1	54.7	11.51
i28	1512.5	36.67	1.18	1717	240.34	7.58	1478.6	47.67	3.31	1542.8	62.8	13.52
i29	1413.1	36.54	1.02	1575	256.3	4.69	1377.9	29.09	3.13	1449.3	54.6	13.23
i30	1524.1	36.20	2.68	1675	238.63	7.96	1467.9	33.18	5.36	1538.3	74.6	14.46
Average	1459.3	38.37	2.11	1611	222.18	6.24	1412.9	36.21	3.84	1472.7	56.29	12.65

Ultimately, it becomes apparent that the overall results achieved in Table 3 have proven that the HSA is promising and competitive, and that it surpasses some other algorithms that have solved the same dataset. The results were also found to be very near to the BKR (the RPD is very close to zero). The favourable results produced by the HSA are testament to the advantage of using the HSA in solving BAP, supporting the above-mentioned hypothesis and obviously answering the query of whether the HSA is able to tackle the BAP efficiently.



(a) The RPD values

(b) The average of the objective function values

(c) The standard deviation values

FIGURE 11. A comparison between HSA, BMO, PSO and GA

5. CONCLUSION AND FUTURE RESEARCH

BAP is one of the renowned challenging combinatorial optimization problems which serves a crucial task in maritime transportation systems. Since BAP is an NP-hard problem, which is difficult to resolve for optimality in a reasonable amount of time, Exact algorithms are not applicable for this kind of large-scale real-world problems. This study looks into the application of HSA for tackling the BAP, as it has never been applied to tackle the BAP until the time of writing. This motivates us to investigate the performance of HSA for this task. The suggested HSA was assessed using (I3) benchmark dataset from the literature and compared with latest algorithms found in the literature to confirm its performance. Experimental results evidenced that the HSA is a promising, competitive, and that it had surpassed some other algorithms that solved the same dataset, other than the fact that the results were very near to the BKR. The favorable results produced by the HSA prove that the HSA is a good alternative in getting high-quality solutions for solving BAP.

Conversely, according to the results presented, the HSA (like other population-based metaheuristics) suffers from slow convergence due to the lack of exploitation (intensification) capabilities. In the future, it is recommended that the HSA is hybridized with single-based metaheuristic for enhancing its performance. Also, it is beneficial if the HSA is applied on different BAP datasets for more consistency. One more potential direction is to study the possibility of solving the HSA on the continuous BAP which is very close to the real-world operations.

FUNDING

None

ACKNOWLEDGEMENT

None

CONFLICTS OF INTEREST

None

REFERENCES

[1] E. T. Bacalhau, L. Casacio, and A. T. de Azevedo, "New hybrid genetic algorithms to solve dynamic berth allocation problem," Expert Systems with Applications, vol. 167, p. 114198, 2021/04/01/ 2021.

[2] T. El Boghdadly, "Evolutionary optimisation approach for the single and multiple-port berth allocations and Quay Crane Assignment Problem," University of Portsmouth, 2018.

[3] F. Rodrigues and A. Agra, "Berth allocation and quay crane assignment/scheduling problem under uncertainty: A survey," European Journal of Operational Research, vol. 303, no. 2, pp. 501-524, 2022/12/01/ 2022.

[4] P. Corry and C. Bierwirth, "The Berth Allocation Problem with Channel Restrictions," Transportation Science, vol. 53, no. 3, pp. 708-727, 2019.

[5] R. Wang, Berth Scheduling at Seaports: Meta-Heuristics and Simulation. Liverpool John Moores University (United Kingdom), 2018.

[6] A. Arram, M. Ayob, and A. Sulaiman, "Hybrid Bird Mating Optimizer With Single-Based Algorithms for Combinatorial Optimization Problems," IEEE Access, vol. 9, pp. 115972-115989, 2021.

[7] C.-J. Ting, K.-C. Wu, and H. Chou, "Particle swarm optimization algorithm for the berth allocation problem," Expert Systems with Applications, vol. 41, no. 4, Part 1, pp. 1543-1550, 2014/03/01/ 2014.

[8] S. Tang, J. G. Jin, and C. Lu, "Investigation of berth allocation problem in container ports considering the variety of disruption," Computers & Industrial Engineering, vol. 172, p. 108564, 2022/10/01/ 2022.

[9] N. Cheimanoff, F. Fontane, M. N. Kitri, and N. Tchernev, "Exact and heuristic methods for the berth allocation problem with multiple continuous quays in tidal bulk terminals," Expert Systems With Applications, vol. 201, p. 117141, 2022.

[10] B. Ahmed, H. Hamdan, A. Muhammed, and N. A. Husin, "Review on the parameter settings in harmony search algorithm applied to combinatorial optimization problems," Indonesian Journal of Electrical Engineering and Computer Science, vol. 27, no. 1, pp. 431-441, 2022.

[11] M. Hadwan, "Annealing Harmony Search Algorithm to Solve the Nurse Rostering Problem," Computers, Materials \& Continua, vol. 71, no. 3, pp. 5545--5559, 2022.

[12] Y.-H. Kim, Y. Yoon, and Z. W. Geem, "A comparison study of harmony search and genetic algorithm for the max-cut problem," Swarm and evolutionary computation, vol. 44, pp. 130-135, 2019.

[13] T. Zhang and Z. W. Geem, "Review of harmony search with respect to algorithm structure," Swarm and Evolutionary Computation, vol. 48, pp. 31-43, 2019/08/01/ 2019.

[14] A. Arram, M. Ayob, G. Kendall, and A. Sulaiman, "Bird Mating Optimizer for Combinatorial Optimization Problems," IEEE Access, vol. 8, pp. 96845-96858, 2020.

[15] J.-F. Cordeau, G. Laporte, P. Legato, and L. Moccia, "Models and Tabu Search Heuristics for the Berth-Allocation Problem," Transportation Science, vol. 39, no. 4, pp. 526-538, 2005/11/01 2005.

[16] L. Guo, J. Zheng, H. Du, J. Du, and Z. Zhu, "The berth assignment and allocation problem considering cooperative liner carriers," Transportation Research Part E: Logistics and Transportation Review, vol. 164, p. 102793, 2022/08/01/ 2022.

[17] G. Zong Woo, K. Joong Hoon, and G. V. Loganathan, "A New Heuristic Optimization Algorithm: Harmony Search," SIMULATION, vol. 76, no. 2, pp. 60-68, 2001/02/01 2001.

[18] A. A. Al-Omoush, A. A. Alsewari, H. S. Alamri, and K. Z. Zamli, "Comprehensive Review of the Development of the Harmony Search Algorithm and its Applications," IEEE Access, vol. 7, pp. 14233-14245, 2019.

[19] M. Dubey, V. Kumar, M. Kaur, and T.-P. Dao, "A Systematic Review on Harmony Search Algorithm: Theory, Literature, and Applications," Mathematical Problems in Engineering, vol. 2021, p. 5594267, 2021/04/16 2021.

[20] E. Abbas and B. Al-Sarray, "Particle Swarm Optimization for Penalize cox models in long-term prediction of breast cancer data," Iraqi Journal For Computer Science and Mathematics, vol. 4, no. 4, pp. 215-224, 2023.

[21] H. Abubakar, A. Muhammad, and S. Bello, "Ants colony optimization algorithm in the Hopfield neural network for agricultural soil fertility reverse analysis," Iraqi Journal For Computer Science and Mathematics, vol. 3, no. 1, pp. 32-42, 2022.