mathematics

Article

Combining a Population-Based Approach with Multiple Linear Models for Continuous and Discrete Optimization Problems

Emanuel Vega [1,*], Ricardo Soto [1], Pablo Contreras [1], Broderick Crawford [1], Javier Peña [1] and Carlos Castro [2]

[1] Escuela de Ingeniería Informática, Pontificia Universidad Católica de Valparaíso, Valparaíso 2362807, Chile
[2] Departamento de Informática, Universidad Técnica Federico Santa María, Valparaíso 2390123, Chile
* Correspondence: emanuel.vega.m@mail.pucv.cl

Abstract: Population-based approaches have given us new search strategies and ideas in order to solve optimization problems. Usually, these methods are based on the performance carried out by a finite number of agents, which by the interaction between them they evolve and work all over the search space. Also, it is well-known that the correct employment of parameter values in this kind of method can positively impact their performance and behavior. In this context, the present work focuses on the design of a hybrid architecture which smartly balances the population size on run-time. In order to smartly balance and control the population size, a modular approach, named Linear Modular Population Balancer (LMPB), is proposed. The main ideas behind the designed architecture include the solving strategy behind a population-based metaheuristic, the influence of learning components based on multiple statistical modeling methods which transform the dynamic data generated into knowledge, and the possibilities to tackle both discrete and continuous optimization problems. In this regard, three modules are proposed for LMPB, which concern tasks such as the management of the population-based algorithm, parameter setting, probabilities, learning methods, and selection mechanism for the population size to employ. In order to test the viability and effectiveness of our proposed approach, we solve a set of well-known benchmark functions and the multidimensional knapsack problem (MKP). Additionally, we illustrate promising solving results, compare them against state-of-the-art methods which have proved to be good options for solving optimization problems, and give solid arguments for future work in the necessity to keep evolving this type of proposed architecture.

Keywords: metaheuristics; machine learning; hybrid approach; optimization

MSC: 90C27; 90C59; 90C15

check for updates

Citation: Vega, E.; Soto, R.; Contreras, P.; Crawford, B.; Peña, J.; Castro, C. Combining a Population-Based Approach with Multiple Linear Models for Continuous and Discrete Optimization Problems. *Mathematics* **2022**, *10*, 2920. https://doi.org/10.3390/math10162920

Academic Editor: Ripon Kumar Chakrabortty

Received: 27 June 2022
Accepted: 8 August 2022
Published: 13 August 2022

1. Introduction

The transformation of data into knowledge has been a trend strategy in modern proposed approaches, they are usually designed by the interdisciplinary interactions of components, such as learning techniques, solving strategies, mathematical ideas, and so on. In this context, data-driven approaches have several objectives, such as identifying key features, identifying redundant data, influencing the decision-making process, and so on [1–3]. In the optimization field, it is well-known that approximated methods try to find solutions as close as possible to the optimum with considerably less usage of resources which has been a trend for years. In this regard, a classic method employed is the Metaheuristics (MH) [4], which are algorithms that follow a pre-designed solving strategy, and can be applied to several optimization problems, and generates massive amount of data in the process [5]. Thus, they have been the objective of multiple works where these attributes are exploited in order to generate knowledge for decision making processes.

In this paper, we propose a novel approach, named Linear Modular Population Balancer (LMPB) as a modular hybrid architecture. We aim to contribute with an optimization tool capable of tackling both discrete and continuous optimization problems, through the interaction of MH and Machine Learning (ML). In this context, LMPB was designed to work under a population-based strategy, which can be seen as a finite number of agents that smartly explore and evolve while searching the solution space. The main objective behind the incorporation of ML into the search process focuses on the fact that population-based MH generates a massive amount of dynamic data through the search in the solution space. Thus, we aim to take advantage of this feature by the means of statistical modeling methods [6–8], take further profit by the knowledge generated, and give adaptability through the search modifying the number of agents which performs on run-time. On the other hand, the proposed LMPB can be described as an interaction of three modules which are described as follows. Firstly, module 1 concerns the management of the search algorithm, carrying out intensification and diversification. In this regard, we employ the movement operators from Spotted Hyena Optimizer (SHO) [9], which is a population-based algorithm, and it has proved to be effective in solving optimization problems [10,11]. Regarding module 2, include multiple tasks concerning internal management in the architecture process. The first task focuses on the management of values employed as population sizes by the architecture, which in the design are presented as schemes. The schemes, correspond to different amounts of agents which can be selected to be employed in a certain period of time. In this context, the selection process is carried out in a Monte Carlo probabilistic roulette mechanism. Thus, at the beginning, each scheme will be assigned an equal probability to be selected, which will be modified in function of the knowledge generated by the learning-based models. This decision behind such a modification in value is based on the objective to achieve a possible improvement in performance in the next period of time. Regarding the second and third tasks, their objective corresponds to the balance of the resulting population size performing the search. The second task is to control the generation of new randomly generated populations in two scenarios: when the selected scheme has a higher number of agents than the one performing and when generating the initial population at the beginning of the search. The third task is in control of removing agents from the population when a newly selected scheme has a smaller population size value than the one currently performing. The last task concerns the management of parameters needed by the proposed architecture to perform, such as the required by the movement operators, probabilities, and learning thresholds employed in the search. Module 3 includes the learning methods designed to process the dynamic data and generate knowledge through the search. This module is based on 6 different learning-based methods which are organized into two groups: 5 statistical modeling methods predicting which scheme has the higher probability to achieve a good performance, and a statistical modeling method selecting which of the 5 mentioned learning methods resultant prognostic should be employed in certain periods of time through the search. This group concerns a single learning method, which is designed by the means of logistic regression. Also, following an equal design, the 5 learning methods are based on Lasso, GammaRegressor, Bayesian, Ridge, and ElasticNet regressions.

In this work, in order to test the viability and the competitiveness of the proposed LMPB, multi-domain experimentation stages are designed. In this context, we solve a set of well-known continuous benchmark functions as a first stage comparison, which are organized as unimodal, multimodal, and multimodal with fixed-dimension, and a set of instances from the multidimensional knapsack problem are solved as a second stage. We analyze, discuss, and compare against reported results from state-of-the-art (SOTA) methods. Moreover, a detailed comparison is carried out by the classic implementation of SHO, Tabu Search (TS) [12], Simulated Annealing (SA) [13], and SHO assisted by IRace implemented by us. We highlight the good performance achieved by the proposed approach, the proper statistical analysis is carried out in order to support the results presented, which proved to be competitive against reported SOTA.

The main contributions can be illustrated as follows.

- Robust hybrid architecture to tackle discrete and continuous optimization problems.

- A key issue in population-based approaches is tackled: Adapting population size on run-time.
- Scalability (module 1), multiple movement operators from different algorithms can be employed in order to carry out intensification and diversification.
- Scalability (modules 3), incorporation of multiple machine learning methods in order o carry out regression and guide the search.

The rest of this paper is organized as follows. The related work is introduced in the next section. The proposed hybrid approach is explained in Section 3. Section 4 illustrates the experimental results. Finally, we conclude and suggest some lines of future research.

2. Related Work

The design of combined optimization tools has been a trend in recent years, the usage of multiple methods has demonstrated to be an effective approach to tackling different issues on the procedures to solve problems [14]. In this context, a well-known example of synergy is the combined usage of optimization techniques and machine learning. They are two fields that are based on artificial intelligence and their interaction has proven great improvements to their respective fields [15,16]. The proposed architecture can be classified as an optimization method assisted by machine learning, where the solving procedure is given by a population-based MH assisted by learning methods.

In the literature, preliminary approaches were designed by the interaction between data mining and evolutionary algorithms [17], the main objective was the analysis of large amounts of data in order to discover patterns, attributes, and so on. The topics developed by this approach have been illustrated as fitness approximations [18], setting parameters [19], initial solutions [20], and population management [21,22]. Regarding this last topic, works were focused on the application of Association rules, where the strategy was to find patterns in elite solutions in order to influence the population and have a higher probability of creating higher quality agents. On the other hand, through the years, a constant evolution has been reported between this interaction [23–25]. For instance, it is well-known that the parameter values employed are highly related to the performance achieved by a MH [26], thus, indispensable components have been developed by the scientific community in order to further improve from this complemented work. In this regard, authors in [27], propose an approach based on Tabu Search (TS) and Support Vector Machine (SVM) in order to successfully solve problems such as Knapsack Problem, Set Covering Problem, and the Travelling Salesman Problem. The general process design includes the decision rules management from a randomly generated corpus of solutions, which are used to predict high-quality solutions for a given instance and it is used to fine-tune and guide the search performed by TS. However, the authors specifically address the proposition as a high complexity approach, a consequence of the design and implementation process in the hybrid, they highlight the time consumed and knowledge necessarily needed, the process to build the corpus, and the extraction of the classification rules. Also, in [28], authors proposed a modular approach in order to tackle the tuning of parameters, where the model iterates by sampling different configurations. The results obtained are used by a regression model, which is based on linear regressions, quantile regression, and ridge or lasso regression, among others. The output of the model is subjected to perturbations resulting in new configuration outputs. Finally, all results obtained are optimized and iteratively tested by the model until a stopping criterion is met. However, a major issue is an exposed consequence of the sampling strategy (usually present in off-line learning) designed in the approach, which is called over-fitting of parameters. In this regard, the proposed LMPB works over an online learning strategy, and all the data is included, classified (for each scheme), and processed.

Regarding hybrids which are related to the population size, to the best of our knowledge, literature is scarce. In [29], authors a cross-entropy-Lagrangian hybrid algorithm for the multi-item capacitated lot-sizing problem. In this proposed approach, response surface methodology is employed to sort the cross entropy parameters values (population size and quantile size) in order to detect a correlation between the assigned values and the heuristic solutions. Also, in [30,31], authors propose a hybrid based on MH assisted by Autonomous Search (AS) in order to modify the population size when stagnation in the performance is detected. In this context, the performance reported from small samples of agents is observed during the search, when there is no better fitness or there is stagnation on the values achieved, the population for Human behavior-based optimization (HBBO) and SHO are modified. This modification on size is static, thus, a predefined amount of agents are added or removed from the population.

3. Proposed Hybrid Approach

In this section, the design of our proposed approach is described and discussed in detail. We illustrate the main ideas behind the proposed modules and learning methods. In Section 3.1 we present a general description of the proposed approach. In Section 3.2, we describe each component concerning, objectives, functionalities, and main ideas. Lastly, we present an overview of the process performed by the architecture in order to carry out the search.

3.1. General Description

The proposed architecture focuses on the balance of the population size on run-time. In order to carry out this objective, specially designed modules, and mechanisms are proposed, Figure 1. The general performance of LMPB, illustrated in Figure 2, has Module 1 at the core, which will be performing a population-based task such as intensification and diversification. Also, all dynamic data generated on run-time will be managed by module 3 which generates knowledge as output that is employed by module 2 in order to carry out key mechanism which gives adaptability in the search. In this context, two mechanisms rule over the search process: the learning mechanism and the population balance mechanism. The employed learning process can be described as a greedy learning mechanism proposed in [11], and its based on constant feedback of knowledge between the learning model and the decision-making component in order to influence/guide the search. The process is as follows, at certain times (a learning season) while carrying out the search, a previously configured threshold value (α) will be defining the seasonal learning on which the dynamic data will be transformed, generating knowledge as feedback to the architecture. This learning strategy is suitable to perform while searching in an unknown solution space, the constant feedback given to LMPB will end up generating a better response through the iterations. This quick and constant knowledge can be a key issue to take into consideration in the design of hybrid approaches in order to keep a continuously smart performance through the search. Regarding the mechanism balancing the population size, the modification process is ruled by the threshold β which defines the number of iterations to perform before carrying out the modification in the number of agents that are performing the search. The configured values are defined as schemes, which are multiple population sizes previously defined. The scheme selection process is ruled by probabilities, which are initially equally assigned values selected by the approach to perform. These probabilities are modified in run-time when threshold α is met and the values employed are based on the knowledge generated in that instance. For instance, the priority of a scheme (higher probability assigned) will be granted based on the best possible prognostic achieved, thus, the component will always search for a better or more fitted configuration in order to improve the performance. In Figure 3 we illustrate a graphic example of how the proposed thresholds are applied through the search.

The general process can be described as follows:

Step 1: Set initial parameters for the population-based method.
Step 2: Set population sizes to be used as schemes.
Step 3: Set initial probabilities to be selected for each scheme.
Step 4: Select a scheme to perform and generate the initial population.
Step 5: Perform SHO: diversification movement operators.

Step 6: All the dynamic data generated in 5 is stored and sorted.
Step 7: Perform SHO: intensification movement operators.
Step 8: All the dynamic data generated in 7 is stored and sorted.
Step 9: if β amount of iterations has been carried out: the selection mechanism will be choosing the next scheme to perform.
Step 10: if α amount of iterations has been carried out: the data is processed, knowledge is generated, and probabilities are updated influenced by the learning-model feedback.
Step 11: if the termination criteria are not met, the search keeps being carried out, return to Step 5.

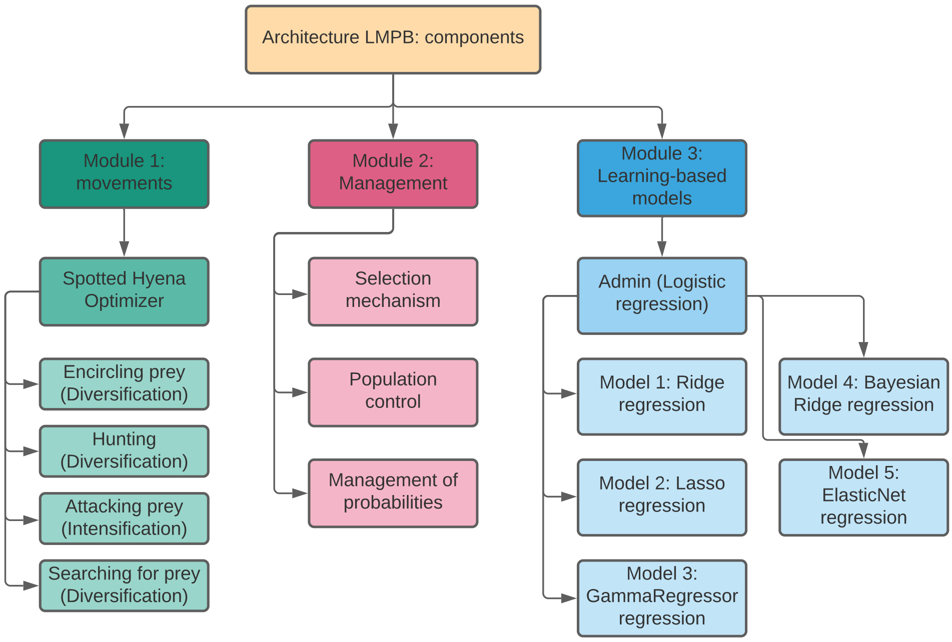

Figure 1. Graphic illustration of the proposed components for LMPB.

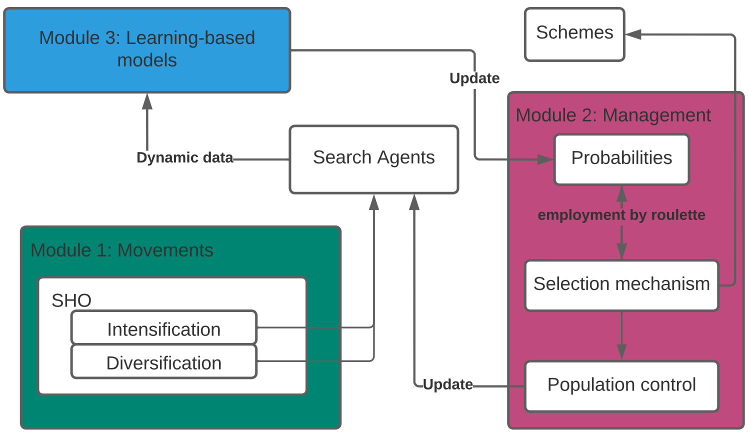

Figure 2. Graphic illustration of the proposed components for LMPB.

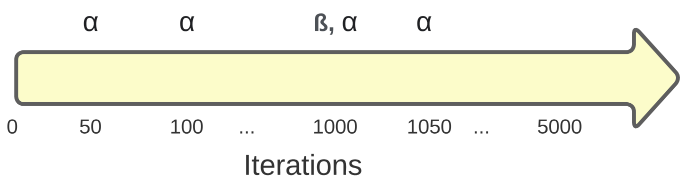

Figure 3. Graphic illustration of the applied thresholds through the search.

3.2. Proposed Modules

As mentioned before, the proposed architecture includes three modules, which are described as follows.

3.2.1. Module 1: Movements

The solving methodology employed in the proposed architecture is, as mentioned before, a population-based strategy. In this regard, multiple agents are generated in order to search the solution space, they evolve through the interaction between the environment and themselves. This interaction is usually defined and structured by the movement operators defined by the algorithm. In this work, we instantiate the SHO algorithm, and employ his four movement operators in order to solve the optimization problems.

Regarding the description from the movement operators, encircling prey is employed first, the objective corresponds to the position update of each agent towards the current best candidate solution (agent with the best solution among the population in that iteration). In order to carry out the perturbation on each agent, we employ Equations (1) and (2). In (1), D_h is the distance between the current agent being updated (P) and the actual best agent in the population (P_p). Also, in Equation (2), each agent is modified (updated). In both equations, B and E correspond to coefficient values, which are illustrated in Equations (3) and (4), where rd_1 and rd_2 are random [0, 1] values. In Equation (5), CI corresponds to the current iteration and TI to the total amount of iterations.

$$D_h = \left| B \cdot P_p(x) - P(x) \right| \tag{1}$$

$$P(x+1) = P_p(x) - E \cdot D_h \tag{2}$$

$$B = 2 \cdot rnd_1 \tag{3}$$

$$E = 2h \cdot rnd_2 - h \tag{4}$$

$$h = 5 - (CI * (5/TI)) \tag{5}$$

The second movement concerns hunting, we employ Equations (6)–(8) in the population. In (6) and (7), D_h represents the distance, P_h represents the actual best agent in the population, P_k is the current agent being updated, and B and E correspond to coefficient values. In (7), and N indicates the number of agents.

$$D_h = |B \cdot P_h - P_k| \tag{6}$$

$$P_k = P_h - E \cdot D_h \tag{7}$$

$$C_h = P_k + P_{k+1} + \cdots + P_{k+N} \tag{8}$$

Attacking the prey is illustrated as the third movement and it is concerned with the performance of exploitation in the search space. In (8), each agent belonging to D_h, generated in (7), will be updated. The last movement exclusively concerns the performance of a passive exploration and is named search for prey. The work proposes the work performed behind coefficients B and E with random values to force the agents to move far away from the actual best agents in the population. This mechanism improves the global search of the approach.

$$P(x+1) = C_h/N \tag{9}$$

3.2.2. Module 2: Management

This module has three main objectives, the first objective concerns population manipulation, where two tasks are performed. Firstly, the elimination of agents from the current population. In this regard, the agents which have the currently worst performance are removed from the population in the scenario where the size of the new selected scheme is smaller. The second task focuses on the addition of new randomly generated agents to the current population; this scenario needs to be carried out when a new scheme is selected, and the size value is bigger. The second objective concerns the scheme selection mechanism, which follows a Monte Carlo roulette strategy, where the main issue is the selection through the probability assigned for each scheme to perform. The population sizes are configured values for each scheme employed by the architecture as illustrated in Table 1. The third objective has a close relationship with objective 2, it is focused on the management of probabilities, as mentioned before, each scheme has a certain probability of being selected and defines the next population size to perform, Table 2. At the beginning, the probabilities are defined as follows.

$$\frac{1}{\text{scheme}_1} + \frac{1}{\text{scheme}_2} + \frac{1}{\text{scheme}_3} + \frac{1}{\text{scheme}_4} = 1$$

where these probabilities will be modified by the output from the learning-based models, the evaluation is carried out as follows.

$$W(scheme_i) = \text{MIN}(y_{scheme_i}, y_{scheme_{i+1}}, \ldots, y_{scheme_n})$$

where $W(scheme_i)$ represents the scheme with the highest possibility to achieve better performance in the next β iterations. For instance, in Table 3 is illustrated a case in which scheme 3 has won and is given a higher probability to be selected.

Table 1. Population sizes employed as schemes.

ID	Amount of Agents
scheme 1	20
scheme 2	30
scheme 3	40
scheme 4	50

Table 2. Probabilities initially assigned to each scheme.

ID	Probability to Be Selected
scheme 1	0.25
scheme 2	0.25
scheme 3	0.25
scheme 4	0.25

Table 3. Modified probabilities for each scheme to be selected.

ID	Probability to be Selected
scheme 1	0.20
scheme 2	0.20
scheme 3	0.40
scheme 4	0.20

3.2.3. Module 3: Learning-Based Methods

The objective behind this module is diagnosis generation, which takes into consideration the performance achieved given the scheme employed. In other words, the general idea is the processing of dynamic data into knowledge, and posterior feedback to module 2. In order to design this module, multiple features were considered, such as the accuracy of the methods, complexity regarding data management, the less expensive (computing time), and implementation complexity. The data transformation process is carried out by the work of 6 different learning methods [8,32], which focus on 2 tasks: administrating and predicting.

Firstly, the objective behind the administrator corresponds to the smart selection of a predictor which aims to decide the most suited regression method to perform when α is met. This method is defined by the means of logistic regression, which despite its name, corresponds to a linear model for classification [33]. The values associated with y, which are the objectives to be predicted, take only small numbers of discrete values, and the fitted function can be illustrated as the following equation.

$$\frac{1}{1+e^z} \tag{10}$$

$$\text{where} \quad z = \beta_0 + \beta_1 x_1 + \cdots + \beta_r x_n$$

Thus, on each learning season (given by the transition of α iterations), this administrator will be carrying out 5 different regressions and deciding which method is more suited/fitted to give a proper prognostic about future performance. Also, the defined dependent variables (x_i) employed can be described as the percentage on which the prognostic output of each method has been employed and the quality of the output given by the accuracy of the prediction. This quality was defined by the percentage of accuracy, which is specified as detecting improvements in the performance when it is successfully selected as the fittest method to perform.

Regarding the predictors, they focus on learning-based methods which aim to give a diagnosis of a possible improved performance given different population configurations (schemes) and performance metrics. They follow the definition of linear models, which can be expressed as a linear combination of multiple features, the fitted function can be described as follows.

$$y(\beta, x) = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n \tag{11}$$

where the classic approach is tackled by the minimization of the residual sum of squares, which can be described as follows.

$$\min_{\text{fi}} \|X\beta - y\|_2^2 \tag{12}$$

$$\text{where} \quad \beta = \beta_0 + \beta_1 + \cdots + \beta_n$$

Firstly, the proposed methods include the employment of Lasso, GammaRegressor, Bayesian, Ridge, and ElasticNet regressions [8]. In this regard, when threshold α is met, each method will be performing over all the schemes configured as illustrated in Figure 4. The dynamic data obtained through the search, such as the feasible/infeasible solution, best solution, and the respective scheme employed which achieved the data are employed to prognostic a possible future fitness value. Thus, every predictor method will have its best prognostic achieved and the final word will be given to the administrator, which decides the scheme to be employed based on the best full prognostic delivered by the predictors.

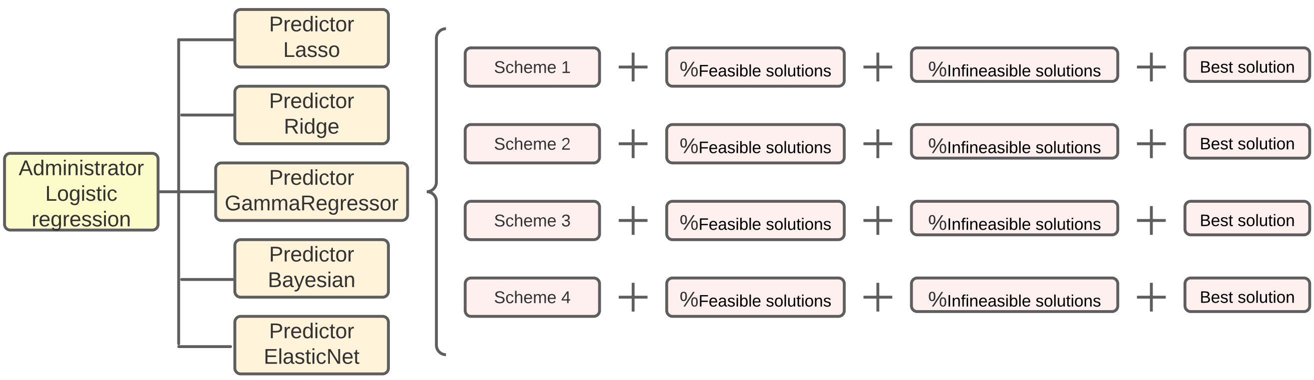

Figure 4. Graphic learning-based models.

The description of each method employed can be described as follows. Lasso and Ridge regressions [34,35] involve adding penalties to the regression functions. They include types of regularization techniques, which are usually used to deal with the over-fitting in the model. Lasso performs L1 regularization, which consists of the addition of a sum of coefficients in the optimization objective. Thus, lasso regression optimizes the following equation.

$$\min_{fi}\left\{\frac{1}{2n_{samples}}\|X\beta-y\|_2^2+\alpha\|\beta\|_1\right\} \tag{13}$$

where the penalty applied corresponds to $\alpha\|\beta\|_1$, with α as a constant value, which can impact the magnitude of the coefficients. Regarding Ridge regression, it performs L2 regularization, which adds a factor of the sum of squares of coefficients in the optimization objective. Thus, the optimization goes as follows.

$$\min_{fi}\left\{\|X\beta-y\|_2^2+\alpha\|\beta\|_2^2\right\} \tag{14}$$

Here, it is important to highlight relevant differences, such as the ridge taking major advantage of the shrinkage of the coefficients, thus, reducing the model complexity and including all or none of the features in the model. Also, Lasso performs a shrinkage of the coefficients and feature selection. GammaRegressor [36] can be included as a generalized linear regression in which a gamma distribution is applied as a probability density function. The generalized linear model can be mathematically described as follows.

$$\min_{fi}\left\{\frac{1}{2n_{samples}}\sum_i d(y_i,\hat{y}_i)+\frac{\alpha}{2}\|\beta\|_2\right\} \tag{15}$$

Here, the gamma distribution defines the target domain $yas(0,\infty)$ where the unit deviance $d(y_i,\hat{y}_i)$ is defined as $2(\log\frac{\hat{y}}{y}+\frac{y}{\hat{y}}-1)$. Regarding ElasticNet [37], is defined as a linear regression that trains with both L1 and L2 regularization in the coefficients. Thus, it is a combination of features from Lasso and Ridge and is fitted to be employed when there are several features correlated between them. The objective function to be minimized is described as follows.

$$\min_{fi}\left\{\frac{1}{2n_{samples}}\|X\beta-y\|_2^2+\alpha p\|\beta\|_1+\frac{\alpha(1-p)}{2}\|\beta\|_2\right\} \tag{16}$$

Lastly, Bayesian linear regression [38] is a statistical analysis that employs Bayesian inference, which is distinguished by the usage of probabilities to express all forms of uncertainty. The main features of this model comprehend the adaptation to the data and give possibilities to add regularization parameters in the statistical work. The probabilistic model can be described as follows.

$$p(y \mid X,w,\alpha)=N(y \mid Xw,\alpha) \tag{17}$$

Here, the output y is assumed to be Gaussian distributed around Xw, and α is manipulated as a stochastic variable that needs to be estimated from the data (disadvantage in the time-consuming inference task).

3.3. Proposed Algorithm

The proposed general search process is illustrated Algorithm 1. In this regard, the solving structure follows a traditional population-based method, where the search is developed under iterative performance, the movement operators of SHO are employed sequentially over each agent in the population. Finally, Algorithm 2 presents the process where the learning components carry out their work.

Algorithm 1 Proposed Architecture

```
SHO: Set initial parameters
Set the size values to perform as schemes
Select a new scheme to perform
Generate initial population based on the scheme selected
while (stooping criteria is not met) do
    SHO: Perform movements operators
    Dynamic data stored and sorted
    if check β amount iterations then
        Select a new scheme to perform
        Balance the population based on the scheme selected
    end if
    if check α amount iterations then
        Call to Algorithm 2: Learning Model
        Check MIN(outputAlgorithm2)
        Data structures with probabilities are updated
    end if
end while
```

Algorithm 2 Learning Model

```
Data processed: percentage of feasible solutions generated over α iterations
Data processed: percentage of infeasible solutions generated over α iterations
Data processed: best solutions generated over α iterations
Performs predictor: lasso
Historical performance data stored and sorted
Performs predictor: ridge
Historical performance data stored and sorted
Performs predictor: gammaregressor
Historical performance data stored and sorted
Performs predictor: bayesian
Historical performance data stored and sorted
Performs Administrator: logistic
Check most suited results to be employed as diagnosis
```

4. Experimental Results

This section illustrates the experimental design and results achieved by the proposed approach. The experimentation is carried out in two phases, solving continuous optimization functions and a well-known discrete optimization problem such as the multidimensional knapsack problem. Thus, each phase describes the optimization problem tackled in detail, and comparison of performance against reported SOTA results. Also, the same configuration of LMPB parameters were employed in both phases, which are illustrated in Table 4.

Table 4. Configuration parameters defined for LMPB.

Parameters	Values
Search agents	Scheme (20, 30, 40, 50)
Control parameter (h)	[5, 0]
M constant	[0.5, 1]
Number of generations	5000
α	50
β	1000

4.1. Continuous Optimization Problem

In this work, in order to test the performance on continuous optimization problems, a set of 15 continuous functions, illustrated in Table 5, are selected to be tackled by LMPB. They are composed of three main categories, such as unimodal [39], multimodal [40], and fixed-dimension multimodal [39,40]. Regarding unimodal functions, they include f_1 to f_4 and correspond to Sphere, Schwefel No.2.22, Schwefel No.1.2, and Generalised Rosenbrock functions. The detailed description is as follows.

Table 5. Optimum values reported for the benchmark functions in the literature, with their corresponding solutions, and search subsets.

Function	Search Subsets	Opt	Sol
$f_1(x)$	$[-100, 100]^{30}$	0	$[0]^{30}$
$f_2(x)$	$[-10, 10]^{30}$	0	$[0]^{30}$
$f_3(x)$	$[-100, 100]^{30}$	0	$[0]^{30}$
$f_4(x)$	$[-30, 30]^{30}$	0	$[1]^{30}$
$f_5(x)$	$[-500, 500]^{30}$	−12596.487	$[420.9687]^{30}$
$f_6(x)$	$[-5.12, 5.12]^{30}$	0	$[0]^{30}$
$f_7(x)$	$[-32, 32]^{30}$	0	$[0]^{30}$
$f_8(x)$	$[-600, 600]^{30}$	0	$[0]^{30}$
$f_9(x)$	$[-50, 50]^{30}$	0	$[1]^{30}$
$f_{10}(x)$	$[-65.536, 65.536]^2$	1	$[-32]^2$
$f_{11}(x)$	$[-5, 5]^2$	−1.0316285	(0.08983, −0.7126) and (−0.08983, 0.7126)
$f_{12}(x)$	$[-5, 10]$ for x_1 and $[0, 15]$ for x_2	0.397887	(−3.142, 12.275), (3.142, 2.275), and (9.425, 2.425)
$f_{13}(x)$	$[-2, 2]^2$	3	(0, −1)
$f_{14}(x)$	$[0, 1]^3$	−3.86	(0.114, 0.556, 0.852)
$f_{15}(x)$	$[0, 1]^6$	−3.32	(0.201, 0.150, 0.477, 0.275, 0.275, 0.377, 0.657)

$$f_1(x) = f(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n} x_i^2 \tag{18}$$

$$f_2(x) = \sum_{i=1}^{n} |x_i| + \prod_{i=1}^{n} |x_i| \tag{19}$$

$$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2 \tag{20}$$

$$f_4(x) = \sum_{i=1}^{n-1} \left[100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2 \right] \tag{21}$$

Regarding multimodal functions, they include f_5 to f_9 and correspond to Generalised Schwefel No.2.26, Generalised Rastrigin, Ackley, Generalised Griewank, and Generalised Penalised Functions. The detailed description is as follows.

$$f_5(x) = -\sum_{i=1}^{n} x_i \sin\left(\sqrt{|x_i|}\right) \tag{22}$$

$$f_6(x) = 10n + \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i)) \tag{23}$$

$$f_7(x) = -20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)) + 20 + \exp(1) \tag{24}$$

$$f_8(x) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) \tag{25}$$

$$f_9(x) = \frac{\pi}{n} \times \left\{ 10\sin^2(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 \left[1 + 10\sin^2(\pi y_{i+1}) \right] + (y_n - 1)^2 \right\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4) \tag{26}$$

where $u(x_i, a, k, m)$ is equal to

1. $k(x_i - a)^m$ if $x_i > a$
2. 0 if $-a \leq x_i \leq a$
3. $k(-x_i - a)^m$ if $x_i < -a$

and

1. $y_i = 1 + \frac{1}{4}(x_i + 1)$

Regarding multimodal functions with fixed-dimension, they include f_{10} to f_{15} and correspond to Shekel's Foxholes, Six-hump Camel Back, Branin, Goldstein-Price, Hartman No.1, and Hartman No.2 functions. The detailed description is as follows.

$$f_{10}(x) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{i,j})^6}\right]^{-1} \tag{27}$$

where:

$$a_{i,j} = \begin{bmatrix} -32 & -16 & 0 & 16 & 32 & -32 & \ldots & 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & \ldots & 32 & 32 & 32 \end{bmatrix}$$

$$f_{11}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4 \tag{28}$$

$$f_{12}(x) = \left(x_2 - \frac{5.1x_1^2}{4\pi^2} + \frac{5x_1}{\pi} - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10 \tag{29}$$

$$\begin{aligned} f_{13}(x) = &\left[1 + (x_1 + x_2 + 1)^2\left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)\right] \\ &\left[30 + (2x_1 - 3x_2)^2\left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)\right] \end{aligned} \tag{30}$$

$$f_{14}(x) = -\sum_{i=1}^{4} c_i \, e^{\left[-\sum_{j=1}^{3} a_{i,j}(x_j - p_{i,j})^2\right]} \tag{31}$$

where the values of a, c, and p are tabulated in Table 6.

Table 6. Values of a_{ij}, c_i, and p_{ij} for function $f_{14}(x)$; n = 3 and j = 1, 2, 3.

i	a_{ij}			c_i	p_{ij}		
1	3	10	30	1	0.3689	0.1170	0.2673
2	0.1	10	35	1.2	0.4699	0.4387	0.7470
3	3	10	30	3	0.1091	0.8732	0.5547
4	0.1	10	30	3.2	0.03815	0.5743	0.8828

$$f_{15}(x) = -\sum_{i=1}^{4} c_i \, e^{\left[-\sum_{j=1}^{6} a_{i,j}(x_j - p_{i,j})^2\right]} \tag{32}$$

where the values of a, c and p are tabulated in Table 7.

Table 7. Values of a_{ij}, c_i, and p_{ij} for function $f_{15}(x)$; n = 6 and $j = 1, 2, \ldots, 6$.

i	a_{ij}						c_i	p_{ij}					
1	10	3	17	3.5	1.7	8	1	0.131	0.169	0.556	0.012	0.828	0.588
2	0.05	10	17	0.1	8	14	1.2	0.232	0.413	0.830	0.373	0.100	0.999
3	3	3.5	1.7	10	17	8	3	0.234	0.141	0.352	0.288	0.304	0.665
4	17	8	0.05	10	0.1	14	3.2	0.404	0.882	0.873	0.574	0.109	0.038

4.1.1. Algorithms Used and Results Comparison

Regarding the results achieved, we carry out multiple comparisons in order to evaluate the current performance, possible short-term improvements, and long-term evolutions in the design. Firstly, in Tables 8–10, we illustrate results reported by SOTA MH, which have proved to reach good performance tackling this set of functions [41–43]. They include particle swarm optimization (PSO) [44], gravitational search algorithm (GSA) [45], differential evolution (DE) [46], whale optimization algorithm (WOA) [41], vapor–liquid equilibrium (VLE) [42], and a specifically designed approach to tackle on this type of benchmark named INMDA, which is a hybrid between Nelder–Mead algorithm and dragonfly algorithm [47]. In this regard, general ideas can be presented, for instance, most standard deviation (StdDev) presented illustrates small values, which can be interpreted as being stagnated in local optima. Also, the proposed INMDA outperforms on most average (Avg) values reported, which presented interesting ideas about the hybridization of stochastic features into an exact method. In this phase, we can observe LMPB achieving competitive results, however, standard deviation (StdDev) computed depicts high values (f_4, f_5, f_6, and f_7) which illustrates potential windows to improvement in the performance, for instance, incrementing the amount of generation for LMPB to work on.

Table 8. Results comparison in unimodal benchmark functions.

F	LMPB		WOA		DE		GSA		PSO		VLE		INMDA	
	Avg	**StdDev**	**Avg**	**StdDev**	**Avg**	**StdDev**	**Avg**	**StdDev**	**Avg**	**StdDev**	**Avg**	**StdDev**	**Avg**	**StdDev**
f_1	0.0907	2.0386	0.0000	0.0000	8.2000×10^{-14}	5.9000×10^{-14}	2.5300×10^{-16}	0.0000	1.3600×10^{-4}	2.0200×10^{-4}	4.4989×10^{-7}	1.413×10^{-6}	0.0000	0.0000
f_2	0.0346	0.5293	0.0000	0.0000	1.5000×10^{-9}	9.9000×10^{-10}	5.5655×10^{-2}	0.1941	4.2144×10^{-2}	4.5421×10^{-2}	3.0840×10^{-6}	6.0498×10^{-6}	0.0000	0.0000
f_3	0.0000	0.0000	5.3900×10^{-7}	2.9300×10^{-6}	6.8000×10^{-11}	7.4000×10^{-11}	8.9353×10^{2}	3.1896×10^{2}	70.126	22.119	5.2020	0.7986	0.0000	0.0000
f_4	28.5342	70.0454	27.866	0.7636	0.0000	0.0000	67.543	62.225	96.718	60.116	79.199	37.400	0.0000	0.0000

Table 9. Results comparison in multimodal benchmark functions.

F	LMPB		WOA		DE		GSA		PSO		VLE		INMDA	
	Avg	**StdDev**	**Avg**	**StdDev**	**Avg**	**StdDev**	**Avg**	**StdDev**	**Avg**	**StdDev**	**Avg**	**StdDev**	**Avg**	**StdDev**
f_5	−914.1975	4974.5174	-5.0808×10^{3}	6.9580×10^{2}	-1.1080×10^{4}	5.7470×10^{2}	-2.8211×10^{3}	4.9304×10^{2}	-4.8413×10^{3}	1.1528×10^{3}	-1.2566×10^{4}	68.705	−2245.1500	2.8400
f_6	0.1865	5.2889	0.0000	0.0000	69.200	38.800	25.968	7.4701	46.704	11.629	34.5830	17.8860	0.0000	0.0000
f_7	7.6581	9.7217	7.4043	9.8976	9.7000×10^{-8}	4.2000×10^{-8}	6.2087×10^{-2}	0.23628	0.27602	0.50901	3.1704	3.9211	0.0000	1.6200×10^{-16}
f_8	0.0056	0.1538	2.8900×10^{-4}	1.5860×10^{-3}	0.0000	0.0000	27.702	5.0403	9.2150×10^{-3}	7.7240×10^{-3}	0.5074	0.5041	0.0000	0.0000
f_9	1.8286	1.5985×10^{-9}	0.3397	0.2149	7.9000×10^{-15}	8.0000×10^{-15}	1.7996	0.95114	6.9170×10^{-3}	2.6301×10^{-2}	0.2369	0.2877	0.0000	0.0000

Table 10. Results comparison in multimodal benchmark functions with fixed-dimension.

F	LMPB		WOA		DE		GSA		PSO		VLE		INMDA	
	Avg	**StdDev**	**Avg**	**StdDev**	**Avg**	**StdDev**	**Avg**	**StdDev**	**Avg**	**StdDev**	**Avg**	**StdDev**	**Avg**	**StdDev**
f_{10}	11.6858	7.8237	2.1120	2.4986	0.99800	3.3000×10^{-16}	5.8598	3.8313	3.6272	2.5608	0.99800	2.5294×10^{-7}	N/A	N/A
f_{11}	0.0001	0.0022	4.2000×10^{-7}	−1.0316	3.1000×10^{-13}	−1.0316	4.8800×10^{-16}	−1.0316	6.2500×10^{-16}	−1.0315	1.8408×10^{-4}	N/A	N/A	
f_{12}	−1.3549	0.2814	0.39791	2.7000×10^{-5}	0.39789	9.9000×10^{-9}	0.39789	0.0000	0.39789	0.0000	0.39815	4.5697×10^{-4}	N/A	N/A
f_{13}	0.0001	0.0022	3.0000	4.2200×10^{-15}	3.0000	2.0000×10^{-15}	3.0000	4.1700×10^{-15}	3.0000	1.3300×10^{-15}	3.0097	1.6256×10^{-2}	N/A	N/A
f_{14}	−1.4299	0.7508	−3.8562	2.7060×10^{-3}	N/A	N/A	−3.8628	2.2900×10^{-15}	−3.8628	2.5800×10^{-15}	−3.8628	6.6880×10^{-5}	N/A	N/A
f_{15}	−0.8621	0.4242	−2.9811	0.37665	N/A	N/A	−3.3178	2.3081×10^{-2}	−3.2663	6.0516×10^{-2}	−3.3179	2.1311×10^{-2}	N/A	N/A

Secondly, in Tables 11 and 12, we illustrate detailed results achieved by the implementations of a hybrid framework that also has been specifically designed to tackle this type of benchmark, named learning-based linear balancer (LB^2) [11] and a classic implementation of SHO assisted by IRace. Regarding the comparison of results, the three approaches presented a competitive performance, however, key elements need to be highlighted and discussed. The proposed LMPB reaches better values (Best) in comparison to LB^2 and SHO-IRace solving the benchmark function. After applying Mann-Whitney, LMPB keeps a difference in functions f_{13} compared to LB^2, and f_3, f_6, f_9, f_{10}, and f_{13} compared to SHO-IRace. In this context, the differences in performance are statistically significant (p-values < 0.05), thus, LMPB is statistically superior in those functions. Also, contrary to the observed StdDev values computed for LB^2, LMPB and SHO-IRace do not fall on constant stagnation (local optima). However, if we observe the average time (Avg time) achieved by the approaches, LB^2 is clearly superior on all the functions solved. This issue can be discussed based on the design and complexity behind both architectures. In this regard, the learning-based component employed by LB^2 was designed by a simple linear model. On the other hand, a more complex design was proposed on LMPB, where multiple linear models are working in parallel, which is mainly the reason for the extra computation time to meet the termination criteria. In this context, new ideas can be proposed as consequence, such as the improvement of the termination criteria by implementing a learning-based component in order to smartly end the search when no possible improvements in the results can be achieved. Lastly, we can confirm that the interaction between machine learning and MH outperforms the classic approach, the idea of profiting through the dynamic data generated can improve the adaptiveness and performance of the methods employed.

4.1.2. Overall Discussion

The comparison against SOTA illustrates that there is no method that is capable of perfectly tackling any optimization problem better than all the others, this also implies that there exists a high difficulty in designing a perfect component for a method in order to keep a suitable balance in the solving strategy for all the optimization problems. On the other hand, after carefully analyzing the results achieved and the performance displayed by the proposed approach to continuous space, positive thoughts about future research are highlighted. Firstly, being able to achieve a competitive performance by tackling the continuous benchmark means that LMPB successfully carried out intensification/diversification and avoided local optima.

- Exploitation analysis: unimodal functions are suitable for benchmarking this issue, the good results achieved can be interpreted that LMPB successfully performed in terms of exploiting optimum values.
- Exploration analysis: multimodal functions are suitable for benchmarking this issue, the competitive performance has proved its merits in terms of exploration and local minima avoidance.

Table 11. Detailed result comparison between the proposed LMPB and LB^2.

F	Opt	LMPB					LB^2				
		Best	**Worst**	**Avg**	**StdDev**	**Avg Time (s)**	**Best**	**Worst**	**Avg**	**StdDev**	**Avg Time (s)**
f_1	0	0	28.2786	0.0907	2.0386	150.1993	0	0	0	0	50.2377
f_2	0	0	14.7092	0.0346	0.5293	190.9703	0	0	0	0	80.7524
f_3	0	0	0	0	0	986.8423	0	0	0	0	96.3627
f_4	0	0	29.4957	28.5342	70.0454	296.1747	1.59197×10^{-7}	1.2262×10^{-6}	6.7549×10^{-7}	5.4204×10^{-7}	71.0024
f_5	−12569.487	−12569.487	9016.3258	−914.1975	4974.5174	250.7817	-1.2570×10^{4}	-1.2567×10^{4}	-1.2569×10^{4}	0.0014	110.3354
f_6	0	0	1.8934	0.1865	1.2189	217.4014	0	0	0	0	60.6482
f_7	0	0	20.0001	7.6581	9.7217	427.1252	4.4408×10^{-16}	4.4409×10^{-16}	4.4409×10^{-16}	0	24.9122
f_8	0	0	7.3880	0.0056	0.1538	255.7067	0	0	0	0	21.7758
f_9	0	1.8290	1.8290	1.8290	0	2223.4575	1.8285	1.8286	1.8286	1.5985×10^{-9}	24.9172
f_{10}	1	6.9407	12.7187	11.6858	7.8237	901.5922	1	1	1	0	17.5661
f_{11}	−1.0316	0	0.0233	0.0001	0.0022	142.0010	0	0	0	0	7.5244
f_{12}	0.3979	−1.1395	−1.5122	−1.3549	0.2814	23.0392	1.1905	2.0325	1.5436	0.4223	4.5528
f_{13}	3	0.0012	0	0.0001	0.0022	129.0010	32.6845	32.6845	32.6845	1.4854×10^{-8}	3.6846
f_{14}	−3.86	−2.0080	−0.0554	−1.4299	0.7508	229.6161	−2.0081	−2.0080	−2.0081	5.0800×10^{-10}	7.1120
f_{15}	−3.32	−1.1676	−0.0056	−0.8621	0.4242	330.3406	−2.1676	−2.1676	−2.1676	0	8.1145

Table 12. Detailed result comparison between the proposed LMPB and SHO-IRace.

F	Opt	LMPB					SHO-IRace				
		Best	**Worst**	**Avg**	**StdDev**	**Avg Time (s)**	**Best**	**Worst**	**Avg**	**StdDev**	**Avg Time (s)**
f_1	0	0	28.2786	0.0907	2.0386	150.1993	0	86.4729	0.1002	2.1974	130.2574
f_2	0	0	14.7092	0.0346	0.5293	190.9703	0	22.2119	0.0362	0.5566	181.1410
f_3	0	0	0	0	0	986.8423	0	2118.0295	97.4849	352.2949	882.2675
f_4	0	0	29.4957	28.5342	70.0454	296.1747	0	188.6322	28.5221	69.3946	271.6308
f_5	−12569.487	−12569.487	9016.3258	−914.1975	4974.5174	250.7817	−12569.4862	9016.3365	−925.5051	4981.0787	229.1431
f_6	0	0	1.8934	0.1865	1.2189	217.4014	0	2382.5545	0.2687	13.1434	163.7028
f_7	0	0	20.0001	7.6581	9.7217	427.1252	4.4408×10^{-16}	22.2358	7.3976	9.6549	325.4619
f_8	0	0	7.3880	0.0056	0.1538	255.7067	0	3.4690	0.0593	0.4755	195.3925
f_9	0	1.8290	1.8290	1.8290	0	2223.4575	35.5837	1766.7315	526.3003	410.1304	2060.7682
f_{10}	1	6.9407	12.7187	11.6858	7.8237	901.5922	12.7186	498.9434	13.1147	9.6306	855.9498
f_{11}	−1.0316	0	0.0233	0.0001	0.0022	142.0010	0	0.1745	0.0001	0.0021	120.5488
f_{12}	0.3979	−1.1395	−1.5122	−1.3549	0.2814	23.0392	−1.1395	−1.6328	−1.4191	0.2372	21.7540
f_{13}	3	0.0012	0	0.0001	0.0022	129.0010	32.6846	635.1801	255.2925	237.1631	204.8439
f_{14}	−3.86	−2.0080	−0.0554	−1.4299	0.7508	229.6161	−2.0080	0.0467	-1.2319	0.7848	183.3399
f_{15}	−3.32	−1.1676	−0.0056	−0.8621	0.4242	330.3406	−2.0080	−1.6155	−0.8480	0.4529	313.5484

It is proved that the interaction of multiple optimization tools brings new possibilities in order to solve hard optimization problems. The complexity at the initial step in defining a design is addressed as an arduous task, the aim is for the selection of certain useful methods (problem-related), identification of potential drawbacks, and the improvement of them by another method. However, knowledge of the features (positive and negative) of every method needs to be clear, thus making it a task for experienced researchers. On the other hand, the incorporation of several methods can bring an increment in the usage of computational resources which is closely related to the design behind the complexity in the framework/architecture. In this experimental test, compared to other approaches, the average solving time was higher and the complexity in the implementation is an issue. In this regard, it is well-known that there is no assurance for techniques to equally perform in different problems/issues, thus, the experimentation with several methods would open new major challenges. Also, in order to tackle the exponential increment in computational time, interesting ideas can be presented such as the improvement in the termination criteria or the employment of sophisticated techniques at the implementation level. Regarding the optimization of continuous problems, two topics will be considered challenging, tackling more complex functions, such as IEEE CEC composite functions and higher dimensional ones, also tackling real-world problems.

4.2. Discrete Optimization Problem

In this work, in order to test the performance of the proposed approach to tackling discrete optimization problems, the Multidimensional Knapsack Problem (MKP) was selected to be solved. In this regard, 6 different instance sets from Beasley's OR library were employed. The details concerning the solved benchmark are illustrated in Table 13.

Table 13. Configuration details from MKP instances employed in this work.

ID	Test Problem	Optimal Solution	n	m
	5.100.00	24381	100	5
	5.100.01	24274	100	5
mknapcb1	5.100.02	23551	100	5
	5.100.03	23534	100	5
	5.100.04	23991	100	5
	5.250.00	59312	250	5
	5.250.01	61472	250	5
mknapcb2	5.250.02	62130	250	5
	5.250.03	59463	250	5
	5.250.04	58951	250	5
	5.500.00	120148	500	5
	5.500.01	117879	500	5
mknapcb3	5.500.02	121131	500	5
	5.500.03	120804	500	5
	5.500.04	122319	500	5
	10.100.00	23064	100	10
	10.100.01	22801	100	10
mknapcb4	10.100.02	22131	100	10
	10.100.03	22772	100	10
	10.100.04	22751	100	10
	10.250.00	59187	250	10
	10.250.01	58781	250	10
mknapcb5	10.250.02	58097	250	10
	10.250.03	61000	250	10
	10.250.04	58092	250	10
	10.500.00	117821	500	10
	10.500.01	119249	500	10
mknapcb6	10.500.02	119215	500	10
	10.500.03	118829	500	10
	10.500.04	116530	500	10

The Multidimensional Knapsack Problem (MKP) can be defined as an NP-hard problem and can be considered the generalized form of the classic Knapsack Problem (KP). The main objective of MKP is to search for a subset of given objects that maximize the total profit while satisfying all constraints on resources. Also, the KP is a well-known optimization problem that has been applied in multiple real-world fields, such as cryptography, allocation problems, scheduling, and production [48,49]. The model can be stated as follows.

$$Maximize \quad \sum_{j=1}^{n} c_j x_j$$

$$Subject \;\; to \quad \sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i \in M = 1, 2, \ldots, m$$

$$x_j \in \{0, 1\}, \quad j \in N = 1, 2, \ldots, n$$

where n is the number of items and m is the number of knapsack constraints with capacities b_i. Each item j requires a_{ij} units of resource consumption in the ith knapsack and yields c_j units of profit upon inclusion. The goal is to find a subset of items that yield maximum profit without exceeding the resource capacities. Additionally, SHO was initially designed to work on a continuous space, in order to tackle the MCDP, SCP, and MKP, a transformation of the domain is needed. In this work, this task is performed by applying binarization strategies, where each strategy is composed of a transfer function [50] and a discretization method. In this regard, we follow the strategy proposed in [51], which employs the transfer function V_4 + Elitist discretization.

4.2.1. Algorithms Used and Results Comparison

Regarding the results achieved, we carry out multiple comparisons in order to evaluate the current performance, possible short-term improvements, and long-term evolutions in the design. Regarding the reported approaches employed to carry out the comparison, they include the filter-and-fan heuristic (F&F) [52], a Binary version of the PSO algorithm (3R-BPSO) [53], and a hybrid quantum particle swarm optimization (QPSO) algorithm [54]. The design behind these methods focuses on solving efficiently the MKP. For instance, the 3R-BPSO algorithm employs three repair operators in order to fix infeasible solutions generated on run-time. Table 14 illustrates the reported performance by the SOTA methods, where the RPD value represents the relative percentage deviation computed as follows.

$$\text{RPD} = \frac{(S - S_{opt})}{S_{opt}} \times 100 \qquad (33)$$

This RPD value will help us to understand the distance between the values reached (Best) against the optimum (Opt) value for each instance. Thus, if we observe the results illustrated, the proposed LMPB achieves equal or better performance in comparison to the SOTA solving the instances mknapcb1, mknapcb2, mknapcb4, and mknapcb5. In Table 15, we illustrate the results achieved by the proposed LMPB vs the implemented SHO assisted by IRace. Regarding the general performance, if we observe the RPD values, the proposed approach is clearly superior achieving 20 optimum values vs 0 reached by SHO-IRace. Also, this can be confirmed after applying Mann-Whitney, which given statistical significance (p-values < 0.05) in the achieved performance tackling the instance 5.100.04 from mknapcb1, all instances from mknapcb2, mknapcb4, and mknapcb5 in comparison to SHO-IRace. In Tables 16–18, we illustrate the results achieved by the proposed LMPB vs the classic implemented SHO, TS, and SA. Regarding the general performance, if we observe the RPD values, the three classic version falls behind in comparison to the proposed approach. Also, this can be confirmed after applying Mann-Whitney, which given statistical significance (p-values < 0.05) in the achieved performance tackling all the instances in comparison to SHO, TS, and SA. We highlight that all approaches do not seem to stagnate in local optima (StdDev), which allows us to understand how solid/well-balanced these proposed methods were initially defined (especially classic methods). Also, SHO, TS, SA, and SHO-IRace reported considerably better solving times in almost all instances in comparison to LMPB. It is a fact that hybrid approaches are leading the current optimization field and are a better answer to complex problems where adaptability in the search space is needed and a key issue to consider in the future. The issue with hybrids is the selection of algorithms, for instance, it is well-known the no-free-lunch theorem has been addressed to MH, where there is no certainty on an MH to achieve an equal performance tackling different kinds of problems. Moreover, an equal situation can be addressed in ML techniques, where the performance is not guaranteed and the complexity between supervised learning vs a deep learning technique is hard to measure.

4.2.2. Overall Discussion

In this experimentation test solving discrete optimization problems, the good overall performance has given us new ideas regarding fully tackling this domain. Firstly, we observed equal phenomena illustrated on the continuous experimentation, competitive performance was achieved against specifically designed approaches. Regarding the performance of SHO, TS, and SA, the achieved results illustrate great deficiencies in comparison to optimum values, also, slightly better values were reached with the assistance of IRace. In addition to all the observations presented in Section 4.1.2, what is interesting to highlight is the change of domain applied to the movements operator of SHO. In the literature, several scientific studies have highlighted the good performance of continuous MH solving discrete optimization problems in comparison to discrete MH [50,51,55]. In this work, we employed the binarization strategy based on V_4 + Elitist discretization, however, several combinations can be tested in order to probably achieve better performance. This binarization issue can be a challenging option to be tackled by a smart component in order to give multiple opportunities in the transformation of the domain to the search in run-time.

Table 14. Computational results achieved by LMPB and state-of-the-art approaches solving the MKP.

ID	Test Problem	Opt	LMPB			QPSO			3R—PSO			F & F		
			Best	Avg	RPD (%)	Best	Avg	RPD (%)	Best	Avg	RPD (%)	Best	Avg	RPD (%)
mknapcb1	5.100.00	24381	24381	18193.2647	0.00	24381	24381	0.00	24381	24381	0.00	24381	N/A	0.00
	5.100.01	24274	24274	17674.1159	0.00	24274	24274	0.00	24274	24274	0.00	24274	N/A	0.00
	5.100.02	23551	23551	17860.9433	0.00	23551	23551	0.00	23538	23538	0.06	23551	N/A	0.00
	5.100.03	23534	23534	19692.4754	0.00	23534	23534	0.00	23534	23508	0.00	23534	N/A	0.00
	5.100.04	23991	23991	17863.3812	0.00	23991	23991	0.00	23991	23961	0.00	23991	N/A	0.00
mknapcb2	5.250.00	59312	59312	46587.9561	0.00	59312	59312	0.00	N/A	N/A	N/A	59312	N/A	0.00
	5.250.01	61472	61472	47299.2074	0.00	61472	61470	0.00	N/A	N/A	N/A	61468	N/A	0.01
	5.250.02	62130	62130	49261.7206	0.00	62130	62130	0.00	N/A	N/A	N/A	62130	N/A	0.00
	5.250.03	59463	59463	46365.1888	0.00	59427	59427	0.06	N/A	N/A	N/A	59436	N/A	0.05
	5.250.04	58951	58951	47005.2385	0.00	58951	58951	0.00	N/A	N/A	N/A	58951	N/A	0.00
mknapcb3	5.500.00	120148	101980	88110.0778	15.12	120130	120105	0.01	120141	102101	0.01	120134	N/A	0.01
	5.500.01	117879	99901	90506.6091	15.25	117844	117834	0.03	117864	117825	0.01	117864	N/A	0.01
	5.500.02	121131	102559	91014.0520	15.33	121112	121092	0.02	121129	121103	0.00	121131	N/A	0.00
	5.500.03	120804	100864	91796.0122	16.50	120804	120740	0.00	120804	120722	0.00	120794	N/A	0.01
	5.500.04	122319	102520	91771.7789	16.18	122319	122300	0.00	122319	122310	0.00	122319	N/A	0.00
mknapcb4	10.100.00	23064	23064	22275.5321	0.00	23064	23064	0.00	23064	23050	0.00	23064	N/A	0.00
	10.100.01	22801	22801	21295.6074	0.00	22801	22801	0.00	22801	22752	0.00	22801	N/A	0.00
	10.100.02	22131	22131	20486.6556	0.00	22131	22131	0.00	22131	22119	0.00	22131	N/A	0.00
	10.100.03	22772	22772	18785.5884	0.00	22772	22772	0.00	22772	22744	0.00	22772	N/A	0.00
	10.100.04	22751	22751	22604.2587	0.00	22751	22751	0.00	22751	22651	0.00	22751	N/A	0.00
mknapcb5	10.250.00	59187	59187	55818.9961	0.00	59182	59173	0.01	N/A	N/A	N/A	59164	N/A	0.04
	10.250.01	58781	58781	55302.6930	0.00	58781	58733	0.00	N/A	N/A	N/A	58693	N/A	0.15
	10.250.02	58097	58097	52907.7982	0.00	58097	58096	0.00	N/A	N/A	N/A	58094	N/A	0.01
	10.250.03	61000	61000	57342.3073	0.00	61000	60986	0.00	N/A	N/A	N/A	60972	N/A	0.05
	10.250.04	58092	58092	55037.2680	0.00	58092	58092	0.00	N/A	N/A	N/A	58092	N/A	0.00
mknapcb6	10.500.00	117821	103226	93309.3655	12.38	117744	117733	0.07	117790	117699	0.03	117734	N/A	0.07
	10.500.01	119249	105088	96823.8780	11.87	119177	119148	0.06	119155	119125	0.08	119181	N/A	0.06
	10.500.02	119215	104870	96151.9076	12.03	119215	119146	0.00	119211	119094	0.00	119194	N/A	0.02
	10.500.03	118829	104308	95338.5665	12.22	118775	118747	0.05	118813	118754	0.01	118784	N/A	0.04
	10.500.04	116530	101380	92260.2844	13.00	116502	116449	0.02	116470	116509	0.05	116471	N/A	0.05

Table 15. Computational results achieved by LMPB and SHO-IRace solving the MKP.

		LMPB						SHO-IRace					
ID	**Opt**	**Best**	**Worst**	**Avg**	**StdDev**	**RPD (%)**	**Avg Time (s)**	**Best**	**Worst**	**Avg**	**StdDev**	**RPD (%)**	**Avg Time (s)**
5.100.00	24381	24381	17595	18193.2647	689.3522	0.00	4669.6999	20661	17595	18269.0889	719.6834	15.25	5872.1652
5.100.01	24274	24274	17401	17674.1159	522.1186	0.00	5697.6984	19792	17401	17680.8992	536.9595	18.46	5553.4974
5.100.02	23551	23551	17692	17860.9433	395.5785	0.00	4292.5007	20119	17692	17956.3902	485.5376	14.57	4467.5135
5.100.03	23534	23534	19685	19692.4754	49.3286	0.00	5347.2370	20703	19685	19692.8931	74.2092	12.02	3854.0709
5.100.04	23991	23991	17744	17863.3812	320.1172	0.00	5747.0107	19525	17744	17840.1698	265.9275	18.61	4897.6560
5.250.00	59312	59312	46049	46587.9561	858.5338	0.00	8670.5223	50256	46049	46612.2596	903.5159	15.26	6656.1823
5.250.01	61472	61472	46890	47299.2074	749.7909	0.00	7810.9763	51527	46890	47277.6690	738.8178	16.17	6568.5947
5.250.02	62130	62130	49237	49261.7206	163.3191	0.00	5671.1701	50292	49237	49257.9839	117.6427	19.05	4843.4766
5.250.03	59463	59463	42804	46365.1888	2137.5436	0.00	16606.7606	50890	42804	46275.6829	2190.8037	14.41	15333.6760
5.250.04	58951	58951	46870	47005.2385	369.0429	0.00	6987.2142	49893	46870	46979.8645	348.9194	15.36	6414.6560
5.500.00	120148	101980	73168	88110.0778	11544.9826	15.12	31594.7054	101400	73168	89634.3614	10969.3236	15.60	40985.2208
5.500.01	117879	99901	71265	90506.6091	11400.2546	15.25	41155.1138	99123	71265	90470.8571	11432.0737	15.91	41596.6308
5.500.02	121131	102559	74678	91014.0520	12735.6287	15.33	33245.1504	103579	74678	94113.1442	11512.8562	14.49	39693.0396
5.500.03	120804	100864	74715	91769.0122	10609.5044	16.50	44675.9107	101572	74715	91395.0128	10851.0576	15.92	39272.9026
5.500.04	122319	102520	74537	91771.7789	10591.1422	16.18	42645.1608	102057	74537	90647.5024	11272.3193	16.56	43738.2077
10.100.00	23064	23064	17298	22275.5321	670.6074	0.00	7179.9602	19751	17298	17766.0012	587.7123	14.36	8278.5790
10.100.01	22801	22801	17352	21295.5074	44.2336	0.00	6618.0995	19081	17352	17470.8750	284.2832	16.31	4660.8592
10.100.02	22131	22131	15699	20486.6556	948.5033	0.00	8081.3328	19342	15699	16531.9192	901.2227	12.60	5975.8820
10.100.03	22772	22772	18817	19795.5884	469.0794	0.00	6866.3064	20017	18817	18861.1892	148.7656	12.09	5070.9132
10.100.04	22751	22751	17564	22604.2587	436.9923	0.00	6945.8575	19667	17564	17804.0787	443.9254	13.55	5626.2527
10.250.00	59187	59187	48086	55818.9961	11675.8756	0.00	9550.5818	52250	48086	48545.8764	815.5280	11.72	7242.5197
10.250.01	58781	58781	43173	55302.6930	5750.7501	0.00	13587.1938	50869	43173	46824.4194	3789.4850	13.46	8701.9378
10.250.02	58097	58097	45538	52907.7982	10827.5062	0.00	15849.1611	50261	45538	46420.7704	1018.7772	13.48	13069.1670
10.250.03	61000	61000	47587	57342.3073	10802.1653	0.00	11107.4894	52286	47587	48855.7066	1996.1527	14.28	6072.3390
10.250.04	58092	58092	47703	55037.2680	11251.2648	0.00	9075.8829	51403	47703	48273.0614	868.3146	11.51	7042.7040
10.500.00	117821	103226	74746	93309.3655	13265.1931	12.38	33763.7203	103608	74746	91656.5522	13723.0371	12.06	30478.6163
10.500.01	119249	105088	76531	96823.8780	12237.0902	11.87	38343.9976	104996	76531	97534.9325	11834.3923	11.95	42585.3414
10.500.02	119215	104870	74620	96151.9076	11857.6879	12.03	46075.8874	105329	74620	95092.7730	12464.6680	11.64	37875.6117
10.500.03	118829	104308	74845	95338.5665	11119.6133	12.22	47983.9497	103663	74845	94803.7957	11431.0257	12.76	43169.6069
10.500.04	116530	101380	74441	92260.2844	10578.1152	13.00	43098.1306	101869	74441	92366.4123	10647.4900	12.58	43326.2896

Table 16. Computational results achieved by LMPB and SHO solving the MKP.

		LMPB						SHO					
ID	**Opt**	**Best**	**Worst**	**Avg**	**StdDev**	**RPD (%)**	**Avg Time (s)**	**Best**	**Worst**	**Avg**	**StdDev**	**RPD (%)**	**Avg Time (s)**
5.100.00	24381	24381	17595	18193.2647	689.3522	0.00	4669.6999	17950	16391	17109.1000	658.0524	26.36	210.4372
5.100.01	24274	24274	17401	17674.1159	522.1186	0.00	5697.6984	17854	16486	17055.0500	541.3379	26.44	181.5390
5.100.02	23551	23551	17692	17860.9433	395.5785	0.00	4292.5007	17886	16256	17297.0000	639.4370	24.05	150.7572
5.100.03	23534	23534	19685	19692.4754	49.3286	0.00	5347.2370	18445	17889	17963.2500	161.9119	21.62	190.4938
5.100.04	23991	23991	17744	17863.3812	320.1172	0.00	5747.0107	17678	17430	17528.4000	105.7115	26.31	140.3210
5.250.00	59312	59312	46049	46587.9561	858.5338	0.00	8670.5223	44891	44453	44596.0000	143.7212	24.31	1230.0471
5.250.01	61472	61472	46890	47299.2074	749.7909	0.00	7810.9763	45928	44306	45047.0000	480.1600	25.28	848.3955
5.250.02	62130	62130	49237	49261.7206	163.3191	0.00	5671.1701	42563	42520	42522.1500	9.3716	31.49	1292.4814
5.250.03	59463	59463	42804	46365.1888	2137.5436	0.00	16606.7606	46782	46038	46257.8500	272.5830	21.32	15333.6760
5.250.04	58951	58951	46870	47005.2385	369.0429	0.00	6987.2142	45445	43815	44565.4000	446.6804	22.91	1076.0837
5.500.00	120148	101980	73168	88110.0778	11544.9826	15.12	31594.7054	91110	89807	90131.7000	417.4011	24.16	2191.6924
5.500.01	117879	99901	71265	90506.6091	11400.2546	15.25	41155.1138	91701	89479	90880.9500	521.3980	22.20	2157.1687
5.500.02	121131	102559	74678	91014.0520	12735.6287	15.33	33245.1504	92436	91702	91753.8500	169.8229	23.68	2873.9049
5.500.03	120804	100864	74715	91769.0122	10609.5044	16.50	44675.9107	93638	91512	93040.6500	487.9807	22.48	2986.8021
5.500.04	122319	102520	74537	91771.7789	10591.1422	16.18	42645.1608	90328	87825	90077.7000	750.9000	26.15	2664.4909
10.100.00	23064	23064	17298	22275.5321	670.6074	0.00	7179.9602	19626	18043	19071.1000	576.5332	14.90	112.5756
10.100.01	22801	22801	17352	21295.5074	44.2336	0.00	6618.0995	17546	16036	17085.5500	377.4207	23.04	99.3756
10.100.02	22131	22131	15699	20486.6556	948.5033	0.00	8081.3328	18057	17012	17309.7000	337.4800	18.40	120.5140
10.100.03	22772	22772	18817	19795.5884	469.0794	0.00	6866.3064	20024	18755	19178.6000	401.2019	12.06	99.6616
10.100.04	22751	22751	17564	22604.2587	436.9923	0.00	6945.8575	18651	18099	18185.0000	171.6164	18.02	97.3623
10.250.00	59187	59187	48086	55818.9961	11675.8756	0.00	9550.5818	45143	44493	44914.5000	310.0302	23.72	566.9247
10.250.01	58781	58781	43173	55302.6930	5750.7501	0.00	13587.1938	48090	47356	47735.0500	297.7889	18.18	590.0571
10.250.02	58097	58097	45538	52907.7982	10827.5062	0.00	15849.1611	47536	45938	47088.3000	421.6500	18.17	492.0703
10.250.03	61000	61000	47587	57342.3073	10802.1653	0.00	11107.4894	47968	46884	47176.6500	330.7825	21.36	589.1354
10.250.04	58092	58092	47703	55037.2680	11251.2648	0.00	9075.8829	47139	44895	46559.7500	854.3255	18.85	933.7649
10.500.00	117821	103226	74746	93309.3655	13265.1931	12.38	33763.7203	90995	89690	90281.8000	381.1162	22.76	2665.9732
10.500.01	119249	105088	76531	96823.8780	12237.0902	11.87	38343.9976	90207	87691	89507.1500	602.0926	24.35	3015.1351
10.500.02	119215	104870	74620	96151.9076	11857.6879	12.03	46075.8874	94196	91359	92369.0500	615.9669	20.98	2569.8929
10.500.03	118829	104308	74845	95338.5665	11119.6133	12.22	47983.9497	94549	91796	93328.1000	614.4442	20.44	2707.8142
10.500.04	116530	101380	74441	92260.2844	10578.1152	13.00	43098.1306	91234	89336	90872.6500	450.8905	21.70	2465.6711

Table 17. Computational results achieved by LMPB and TS solving the MKP.

		LMPB						TS					
ID	**Opt**	**Best**	**Worst**	**Avg**	**StdDev**	**RPD (%)**	**Avg Time (s)**	**Best**	**Worst**	**Avg**	**StdDev**	**RPD (%)**	**Avg Time (s)**
5.100.00	24381	24381	17595	18193.2647	689.3522	0.00	4669.6999	17920	14646	17200.2660	441.6150	26.50	9.2408
5.100.01	24274	24274	17401	17674.1159	522.1185	0.00	5697.6984	17895	15281	17209.6480	764.0746	26.28	8.1997
5.100.02	23551	23551	17692	17860.9432	395.5784	0.00	4292.5006	17557	15473	16471.3200	537.5266	25.45	9.0440
5.100.03	23534	23534	19685	19692.4753	49.3286	0.00	5347.2370	18153	14953	18068.7160	764.7306	22.86	6.9277
5.100.04	23991	23991	17744	17863.3811	320.1171	0.00	5747.0107	17599	15722	17760.1780	245.7550	26.64	8.1933
5.250.00	59312	59312	46049	46587.9560	858.5338	0.00	8670.5223	45431	41916	45392.1620	297.6204	23.40	51.9413
5.250.01	61472	61472	46890	47299.2073	749.7908	0.00	7810.9763	44651	39048	42666.3920	1629.9833	27.36	90.8240
5.250.02	62130	62130	49237	49261.7205	163.3190	0.00	5671.1700	44587	42244	43400.5440	710.5724	28.24	55.6142
5.250.03	59463	59463	42804	46365.1887	2137.5435	0.00	16606.7606	46510	40376	46108.0680	1191.6083	21.78	73.4146
5.250.04	58951	58951	46870	47005.2385	369.0429	0.00	6987.2141	43622	41511	43578.5660	235.7575	26.00	83.7176
5.500.00	120148	101980	73168	88110.0777	11544.9826	15.12	31594.7054	89365	85199	88040.6580	1055.1352	25.62	657.7096
5.500.01	117879	99901	71265	90506.6090	11400.2546	15.25	41155.1138	91192	87326	90738.1740	1051.4631	22.64	380.6708
5.500.02	121131	102559	74678	91014.0520	12735.6287	15.33	33245.1504	92155	87168	90280.1500	2329.0822	23.92	448.3687
5.500.03	120804	100864	74715	91769.0122	10609.5044	16.50	44675.9107	92344	88444	91129.6820	993.0370	23.56	417.6390
5.500.04	122319	102520	74537	91771.7788	10591.1422	16.18	42645.1608	86955	80832	85634.6120	985.2763	28.91	326.0826
10.100.00	23064	23064	17298	32275.5320	24670.6074	0.00	7179.9601	19365	17117	19292.6880	607.9159	16.04	4.2649
10.100.01	22801	22801	17352	31295.5074	24044.2336	0.00	6618.0994	18535	16420	17955.4980	714.6238	18.71	5.3840
10.100.02	22131	22131	15699	30486.6555	23948.5033	0.00	8081.3327	17523	14835	16785.3360	484.2500	20.82	3.3187
10.100.03	22772	22772	18817	32795.5884	24469.0794	0.00	6866.3063	18229	18179	18190.5000	21.0416	19.95	4.3792
10.100.04	22751	22751	17564	32604.2586	24436.9923	0.00	6945.8575	18833	17619	18463.4220	277.1007	17.22	4.9251
10.250.00	59187	59187	48086	55818.9960	11675.8756	0.00	9550.5818	44135	40025	43711.1320	933.0496	25.43	39.0509
10.250.01	58781	58781	43173	55302.6930	10750.7501	0.00	13587.1938	46438	42427	45226.7400	943.0535	21.00	47.0854
10.250.02	58097	58097	45538	52907.7982	10827.5062	0.00	15849.1611	44080	41890	43428.4520	463.4027	24.13	40.9750
10.250.03	61000	61000	47587	57342.3073	10802.1653	0.00	11107.4894	46377	45074	46255.3360	258.9354	23.97	43.8730
10.250.04	58092	58092	47703	55037.2679	11251.2648	0.00	9075.8829	43049	38232	42366.8760	981.8458	25.90	42.2721
10.500.00	117821	103226	74746	93309.3654	13265.1931	12.38	33763.7203	90919	89123	90331.2340	447.7161	22.83	178.1260
10.500.01	119249	105088	76531	96823.8780	12237.0902	11.87	38343.9976	91968	85869	91923.8820	1797.3181	22.88	231.8760
10.500.02	119215	104870	74620	96151.9075	11857.6879	12.03	46075.8874	95984	92567	95468.6580	1811.0270	19.49	270.4680
10.500.03	118829	104308	74845	95338.5664	11119.6133	12.22	47983.9497	91297	85080	90911.8560	1131.8605	23.17	197.9920
10.500.04	116530	101380	74441	92260.2844	10578.1152	13.00	43098.1306	92792	88027	93015.9780	1423.4816	20.37	342.9408

Table 18. Computational results achieved by LMPB and SA solving the MKP.

		LMPB						SA					
ID	**Opt**	**Best**	**Worst**	**Avg**	**StdDev**	**RPD (%)**	**Avg Time (s)**	**Best**	**Worst**	**Avg**	**StdDev**	**RPD (%)**	**Avg Time (s)**
5.100.00	24381	24381	17595	18193.2647	689.3522	0.00	4669.6999	16645	15750	16488.2857	257.5163	31.73	15.5006
5.100.01	24274	24274	17401	17674.1159	522.1185	0.00	5697.6984	16732	15574	16061.4381	560.7617	31.07	15.0110
5.100.02	23551	23551	17692	17860.9432	395.5784	0.00	4292.5006	14663	13380	14398.9333	378.2078	37.74	9.1421
5.100.03	23534	23534	19685	19692.4753	49.3286	0.00	5347.2370	17033	14747	16594.0540	730.5726	27.62	10.4877
5.100.04	23991	23991	17744	17863.3811	320.1171	0.00	5747.0107	17106	16307	16974.1016	296.6305	28.70	12.3591
5.250.00	59312	59312	46049	46587.9560	858.5338	0.00	8670.5223	44861	43230	44563.9048	518.9806	24.36	76.2560
5.250.01	61472	61472	46890	47299.2073	749.7908	0.00	7810.9763	41902	41321	41646.1333	249.8855	31.84	65.4760
5.250.02	62130	62130	49237	49261.7205	163.3190	0.00	5671.1700	43316	40636	42807.8381	791.1798	30.28	54.3458
5.250.03	59463	59463	42804	46365.1887	2137.5435	0.00	16606.7606	48112	41941	46223.6159	2115.7624	19.09	72.0230
5.250.04	58951	58951	46870	47005.2385	369.0429	0.00	6987.2141	44235	42284	44005.2921	447.8486	24.96	91.7240
5.500.00	120148	101980	73168	88110.0777	11544.9826	15.12	31594.7054	91226	87931	90928.0222	669.9465	24.07	333.4513
5.500.01	117879	99901	71265	90506.6090	11400.2546	15.25	41155.1138	90749	88213	90514.8825	512.7554	23.02	365.0060
5.500.02	121131	102559	74678	91014.0520	12735.6287	15.33	33245.1504	88397	86003	87795.4984	701.4480	27.02	219.4263
5.500.03	120804	100864	74715	91769.0122	10609.5044	16.50	44675.9107	89615	88855	89479.8889	290.5674	25.82	289.9401
5.500.04	122319	102520	74537	91771.7788	10591.1422	16.18	42645.1608	87974	84700	87449.4000	952.0907	28.08	393.7943
10.100.00	23064	23064	17298	32275.5320	24670.6074	0.00	7179.9601	18645	17245	18052.5873	629.0033	19.16	7.8250
10.100.01	22801	22801	17352	31295.5074	24044.2336	0.00	6618.0994	18841	17515	18615.5016	423.9138	17.37	10.6280
10.100.02	22131	22131	15699	30486.6555	23948.5033	0.00	8081.3327	17465	16575	17456.6349	70.7870	21.08	8.9347
10.100.03	22772	22772	18817	32795.5884	24469.0794	0.00	6866.3063	18152	15786	17972.5873	402.4176	20.29	8.8359
10.100.04	22751	22751	17564	32604.2586	24436.9923	0.00	6945.8575	18705	17431	18372.9206	553.1250	17.78	8.9342
10.250.00	59187	59187	48086	55818.9960	11675.8756	0.00	9550.5818	43280	39946	42544.8889	940.8290	26.88	31.3592
10.250.01	58781	58781	43173	55302.6930	10750.7501	0.00	13587.1938	46785	43999	46371.5143	752.3758	20.41	56.4902
10.250.02	58097	58097	45538	52907.7982	10827.5062	0.00	15849.1611	43558	42288	43386.3968	320.3015	25.03	46.1420
10.250.03	61000	61000	47587	57342.3073	10802.1653	0.00	11107.4894	42822	40426	42461.4381	766.1158	29.80	51.4840
10.250.04	58092	58092	47703	55037.2679	11251.2648	0.00	9075.8829	41685	40537	41224.6794	456.9968	28.24	82.5832
10.500.00	117821	103226	74746	93309.3654	13265.1931	12.38	33763.7203	90741	87278	90169.9238	720.1538	22.98	139.6827
10.500.01	119249	105088	76531	96823.8780	12237.0902	11.87	38343.9976	89316	87726	89057.6190	400.3709	25.10	171.5081
10.500.02	119215	104870	74620	96151.9075	11857.6879	12.03	46075.8874	91262	89985	91043.4254	445.0307	23.45	205.4551
10.500.03	118829	104308	74845	95338.5664	11119.6133	12.22	47983.9497	90655	89157	90110.2825	511.5386	23.71	175.9941
10.500.04	116530	101380	74441	92260.2844	10578.1152	13.00	43098.1306	91587	88839	91338.3778	528.3976	21.40	349.4433

5. Conclusions and Future Work

In this paper, a competitive learning-based architecture is proposed, and well-known methods and techniques are employed to design a novel hybrid approach capable of tackling discrete and continuous optimization problems. The main objective behind the proposed design is the interaction between MH and machine learning, where LMPB follows a population-based solving strategy assisted by multiple linear models that profit from the dynamic data generated on run-time. Regarding the performance observed through the experimentation phase, LMPB achieved competitive results tackling both discrete and continuous optimization problems. In this regard, the proposed architecture went against specially designed methods which have proved to perform on such problems, while LMPB employed a unique configuration set of parameters for both cases, which makes the development of this approach an attractive topic and worth researching. Nevertheless, it is important to highlight issues observed in the testing which can be potential paths to carry out future improvements. Firstly, the complexity implementing the architecture can be described in two topics: MH algorithm and learning method employed. In this first attempt proposing LMPB, we instantiate SHO as a potential alternative, however, it is possible to instantiate multiple algorithms in order to define a more complex component of the architecture. Also, the learning model employed is a key issue, which impacts the solving time needed to meet the termination criteria. In this regard, the linear model proved to work for LMPB, however, several learning methods aim for regression. Thus, multiple experiments need to be carried out in order to find better options in order to improve the performance and adaptiveness of the architecture. Concerning the increment in solving time, the complexity behind the architecture and the mechanism employed are the key issue. Thus, as the results improve, it is worth working on the improvement of this optimization issue (termination criteria). Regarding future scope, the focus is on improving modules 1 and 3. In module 1 we want to implement a new population-based MH in order to have more options for applying intensification and diversification. for instance, a possible idea is illustrated in Figure 5, where module 1 will be managing two big groups of movements operators from SHO, Crow Search Algorithm (CSA), and Shuffle Frog Leaping Algorithm (SFLA) which are modern population MH. On the other hand, as mentioned in Section 4.2.2, add the capability to try several binarization strategies in order to smartly guide the transformation of the domain in the variables. In module 3, the aim is to implement other regression methods, such as SVM, Deep learning approaches, and so on. The final objective is to have rich adaptability given the most fitted method to perform prognostic on run-time.

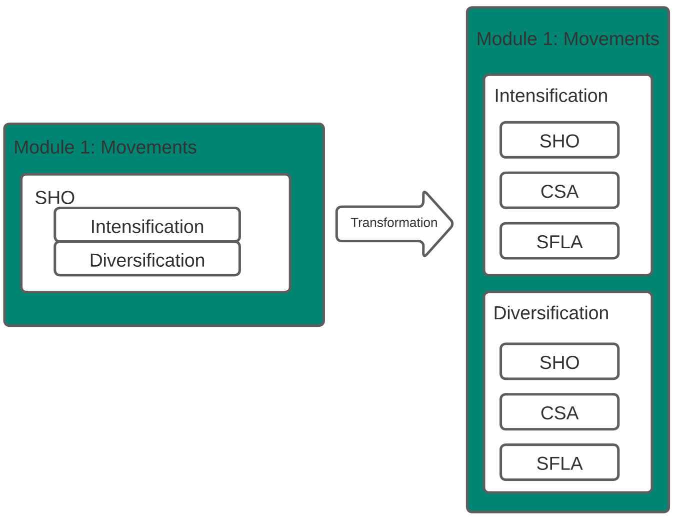

Figure 5. Graphic illustration of the proposed improvement to be carried out in module 1.

Author Contributions: Formal analysis, E.V., P.C., J.P. and R.S.; investigation, E.V., P.C., J.P., R.S., B.C. and C.C.; resource, R.S.; software, E.V. and P.C.; validation, B.C. and C.C.; writing–original draft, E.V., R.S. and B.C.; writing—review and editing, E.V. and R.S. All authors have read and agreed to the published version of the manuscript.

Funding: Ricardo Soto is supported by Grant CONICYT/FONDECYT/REGULAR/1190129. Broderick Crawford is supported by Grant ANID/FONDECYT/REGULAR/1210810, and Emanuel Vega is supported by National Agency for Research and Development ANID/Scholarship Program/DOCTORADO NACIONAL/2020-21202527.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: No new data were created or analysed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The authors declare no conflict of interest. The founding sponsors had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, and in the decision to publish the results.

References

1. Yang, Y.; Zhang, Y.; Meng, X. A data-driven approach for optimizing the EV charging stations network. *IEEE Access* **2020**, *8*, 118572–118592. [CrossRef]
2. Wu, Z.; Hu, J.; Ai, X.; Yang, G. Data-driven approaches for optimizing EV aggregator power profile in energy and reserve market. *Int. J. Electr. Power Energy Syst.* **2021**, *129*, 106808. [CrossRef]
3. Wei, Y.; Zhang, X.; Shi, Y.; Xia, L.; Pan, S.; Wu, J.; Han, M.; Zhao, X. A review of data-driven approaches for prediction and classification of building energy consumption. *Renew. Sustain. Energy Rev.* **2018**, *82*, 1027–1047. [CrossRef]
4. Khajehzadeh, M.; Taha, M.R.; El-Shafie, A.; Eslami, M. A survey on meta-heuristic global optimization algorithms. *Res. J. Appl. Sci. Eng. Technol.* **2011**, *3*, 569–578.
5. Stork, J.; Eiben, A.E.; Bartz-Beielstein, T. A new taxonomy of global optimization algorithms. *Nat. Comput*. **2020**, *21* , 1–24. [CrossRef]
6. Searle, S.R.; Gruber, M.H. *Linear Models*; John Wiley & Sons: Hoboken, NJ, USA, 2016.
7. Hastie, T.J.; Pregibon, D. Generalized linear models. In *Statistical Models in S*; Routledge: London, UK, 2017; pp. 195–247.
8. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
9. Dhiman, G.; Kumar, V. Spotted hyena optimizer: A novel bio-inspired based metaheuristic technique for engineering applications. *Adv. Eng. Softw*. **2017**, *114*, 48–70. [CrossRef]
10. Luo, Q.; Li, J.; Zhou, Y.; Liao, L. Using spotted hyena optimizer for training feedforward neural networks. *Cogn. Syst. Res.* **2021**, *65*, 1–16 [CrossRef]
11. Vega, E.; Soto, R.; Crawford, B.; Peña, J.; Castro, C. A learning-based hybrid framework for dynamic balancing of exploration-exploitation: Combining regression analysis and metaheuristics. *Mathematics* **2021**, *9*, 1976. [CrossRef]
12. Glover, F. Future paths for integer programming and links to artificial intelligence. *Comput. Oper. Res*. **1986**, *13*, 533–549. [CrossRef]
13. Kirkpatrick, S. Optimization by simulated annealing: Quantitative studies. *J. Stat. Phys.* **1984**, *34*, 975–986. [CrossRef]
14. Talbi, E.G. Combining metaheuristics with mathematical programming, constraint programming and machine learning. *Ann. Oper. Res*. **2016**, *240*, 171–215. [CrossRef]
15. Song, H.; Triguero, I.; Özcan, E. A review on the self and dual interactions between machine learning and optimisation. *Prog. Artif. Intell*. **2019**, *8*, 143–165. [CrossRef]
16. Talbi, E.G. Machine learning into metaheuristics: A survey and taxonomy. *ACM Comput. Surv.* **2021**, *54*, 1–32. [CrossRef]
17. Jourdan, L.; Dhaenens, C.; Talbi, E.G. Using datamining techniques to help metaheuristics: A short survey. In *International Workshop on Hybrid Metaheuristics*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 57–69.
18. Jin, Y. A comprehensive survey of fitness approximation in evolutionary computation. *Soft Comput*. **2005**, *9*, 3–12. [CrossRef]
19. Hong, T.P.; Wang, H.S.; Chen, W.C. Simultaneously applying multiple mutation operators in genetic algorithms. *J. Heuristics* **2000**, *6*, 439–455. [CrossRef]
20. Ramsey, C.L.; Grefenstette, J.J. Case-Based Initialization of Genetic Algorithms. In Proceedings of the 5th International Conference on Genetic Algorithms, Urbana-Champaign, IL, USA, 1 June 1993; pp. 84–91.
21. Dalboni, F.L.; Ochi, L.S.; Drummond, L.M. A. On improving evolutionary algorithms by using data mining for the oil collector vehicle routing problem. In Proceedings of the International Network Optimization Conference, Evry/Paris, France, 27–29 October 2003; pp. 182–188.
22. Santos, H.G.; Ochi, L.S.; Marinho, E.H.; Drummond, L.M.D.A. Combining an evolutionary algorithm with data mining to solve a single-vehicle routing problem. *Neurocomputing* **2006**, *70*, 70–77. [CrossRef]

23. Calvet, L.; de Armas, J.; Masip, D.; Juan, A.A. Learnheuristics: Hybridizing metaheuristics with machine learning for optimization with dynamic inputs. *Open Math.* **2017**, *15*, 261–280. [CrossRef]
24. Jong, K.D. Parameter setting in EAs: A 30 year perspective. In *Parameter Setting in Evolutionary Algorithms*; Springer: Berlin/Heidelberg, Gemrany, 2007; pp. 1–18.
25. Karimi-Mamaghan, M.; Mohammadi, M.; Meyer, P.; Karimi-Mamaghan, A.M.; Talbi, E.G. Machine Learning at the service of Meta-heuristics for solving Combinatorial Optimization Problems: A state-of-the-art. *Eur. J. Oper. Res.* **2022**, *296*, 393–422. [CrossRef]
26. Talbi, E.G. *Metaheuristics: From Design to Implementation*; John Wiley & Sons: Hoboken, NJ, USA, 2009; Volume 74.
27. Zennaki, M.; Ech-Cherif, A. A new machine learning based approach for tuning metaheuristics for the solution of hard combinatorial optimization problems. *J. Appl. Sci.* **2010**, *10*, 1991–2000. [CrossRef]
28. Trindade, Á.R.; Campelo, F. Tuning metaheuristics by sequential optimisation of regression models. *Appl. Soft Comput.* **2019**, *85*, 105829. [CrossRef]
29. Caserta, M.; Rico, E.Q. A cross entropy-Lagrangean hybrid algorithm for the multi-item capacitated lot-sizing problem with setup times. *Comput. Oper. Res.* **2009**, *36*, 530–548. [CrossRef]
30. Soto R.; Crawford B.; Vega E.; Gómez A.; Gómez-Pulido J.A. Solving the Set Covering Problem Using Spotted Hyena Optimizer and Autonomous Search. In *Advances and Trends in Artificial Intelligence. From Theory to Practice*; IEA/AIE 2019; Wotawa F., Friedrich G., Pill I., Koitz-Hristov, R., Ali M., Eds.; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2019; Volume 11606.
31. Soto, R.; Crawford, B.; González, F.; Vega, E.; Castro, C.; Paredes, F. Solving the Manufacturing Cell Design Problem Using Human BehaviorBased Algorithm Supported by Autonomous Search. *IEEE Access* **2019**, *7*, 132228–132239. [CrossRef]
32. Egwim, C.N.; Egunjobi, O.O.; Gomes, A.; Alaka, H. A Comparative Study on Machine Learning Algorithms for Assessing Energy Efficiency of Buildings. In Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Online, 13–17 September 2021; Springer: Cham, Switzerland, 2021; pp. 546–566.
33. Menard, S. *Applied Logistic Regression Analysis*; Sage: Newcastle upon Tyne, UK, 2002; Volume 106.
34. Tibshirani, R. Regression shrinkage and selection via the lasso. *J. R. Stat. Soc. Ser.* **1996**, *58*, 267–288. [CrossRef]
35. McDonald, G.C. Ridge regression. *Wiley Interdiscip. Rev. Comput. Stat.* **2009**, *1*, 93–100. [CrossRef]
36. Akwimbi, J. Modelling The Growth of Pension Funds Using Generalized Linear Model (gamma Regression). Ph.D. Thesis, University of Nairobi, Nairobi, Kenya, 2014.
37. Yu, L.; Ma, X.; Wu, W.; Wang, Y.; Zeng, B. A novel elastic net-based NGBMC (1, n) model with multi-objective optimization for nonlinear time series forecasting. *Commun. Nonlinear Sci. Numer. Simul.* **2021**, *96*, 105696. [CrossRef]
38. Gelman, A.; Carlin, J.B.; Stern, H.S.; Rubin, D.B. *Bayesian Data Analysis*; Chapman and Hall/CRC: Boca Raton, FL, USA, 1995.
39. Digalakis, J.; Margaritis, K. On benchmarking functions for genetic algorithms. *Int. J. Comput. Math* **2001**, *77*, 481–506. [CrossRef]
40. Yang, X. Firefly algorithm, stochastic test functions and design optimisation. *Int. J. Bio-Inspired Comput.* **2010**, *2*, 78–84. [CrossRef]
41. Mirjalili, S.; Lewis, A. The Whale Optimization Algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]
42. Cortés-Toro, E.M.; Crawford, B.; Gómez-Pulido, J.A.; Soto, R.; Lanza-Gutiérrez, J.M. A New Metaheuristic Inspired by the Vapour-Liquid Equilibrium for Continuous Optimization. *Appl. Sci.* **2018**, *8*, 2080. [CrossRef]
43. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey Wolf Optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
44. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the IEEE International Conference on Neural Networks, Perth, Australia, 27 November–1 December 1995; pp. 1942–1948.
45. Rashedi, E.; Nezamabadi-Pour, H.; Saryazdi, S. GSA: A Gravitational Search Algorithm. *Inf. Sci.* **2009**, *179*, 2232–2248. [CrossRef]
46. Storn, R.; Price, K. Differential Evolution—A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
47. Xu, J.; Yan, F. Hybrid Nelder–Mead algorithm and dragonfly algorithm for function optimization and the training of a multilayer perceptron. *Arab. J. Sci. Eng.* **2019**, *44*, 3473–3487. [CrossRef]
48. Pisinger, D. The quadratic knapsack problem—A survey. Discrete applied mathematics. *Discret. Appl. Math.* **2007**, *155*, 623–648. [CrossRef]
49. Horowitz, E.; Sahni, S. Computing partitions with applications to the knapsack problem. *J. ACM* **1974**, *21*, 277–292. [CrossRef]
50. Mirjalili, S.; Lewis, A. S-shaped versus v-shaped transfer functions for binary particle swarm optimization. *Swarm Evol. Comput.* **2013**, *9*, 1–14. [CrossRef]
51. Lanza-Gutierrez, J.M.; Crawford, B.; Soto, R.; Berrios, N.; Gomez-Pulido, J.A.; Paredes, F. Analyzing the effects of binarization techniques when solving the set covering problem through swarm optimization. *Expert Syst. Appl.* **2017**, *70*, 67–82. [CrossRef]
52. Khemakhem, M.; Haddar, B.; Chebil, K.; Hanafi, S. A Filter-and-Fan Metaheuristic for the 0–1 Multidimensional Knapsack Problem. *Int. J. Appl. Metaheuristic Comput.* **2012**, *3*, 43–63. [CrossRef]
53. Chih, M. Three pseudo-utility ratio-inspired particle swarm optimization with local search for multidimensional knapsack problem. *Swarm Evol. Comput.* **2018**, *39*, 279–296. [CrossRef]
54. Haddar, B.; Khemakhem, M.; Hanafi, S.; Wilbaut, C. A hybrid quantum particle swarm optimization for the multidimensional knapsack problem. *Eng. Appl. Artif. Intell.* **2016**, *55*, 1–13. [CrossRef]
55. Lemus-Romani, J.; Becerra-Rozas, M.; Crawford, B.; Soto, R.; Cisternas-Caneo, F.; Vega, E.; García, J. A novel learning-based binarization scheme selector for swarm algorithms solving combinatorial problems. *Mathematics* **2021**, *9*, 2887. [CrossRef]