12-4-2009

A hybrid extremal optimisation approach for the bin packing problem

Pedro Gomez-Meneses

Marcus Randall
Bond University, marcus_randall@bond.edu.au

A Hybrid Extremal Optimisation Approach for the Bin Packing Problem

Pedro Gómez-Meneses[1,2] and Marcus Randall[1]

[1] School of Information Technology, Bond University, QLD 4229, Australia
pedgomez@bond.edu.au, mrandall@bond.edu.au
[2] Universidad Católica de la Santísima Concepción, Concepción, Chile
pgomez@ucsc.cl

Abstract. Extremal optimisation (EO) is a simple and effective technique that is influenced by nature and which is especially suitable to solve assignment type problems. EO uses the principle of eliminating the weakest or the least adapted component and replacing it by a random one. This paper presents a new hybrid EO approach that consists of an EO framework with an improved local search for the bin packing problem (BPP). The stochastic nature of the EO framework allows the solution to move between feasible and infeasible spaces. Hence the solution has the possibility of escaping from a stagnant position to explore new feasible regions. The exploration of a feasible space is complemented with an improved local search mechanism developed on the basis of the proposed Falkenauer's technique. The new local search procedure increases the probability of finding better solutions. The results show that the new algorithm is able to obtain optimal and efficient results for large problems when the approach is compared with the best known methods.

1 Introduction

In recent times, a large amount of research has been undertaken to explore novel optimisation meta-heuristics inspired by nature that can solve complex problems. These techniques seek to use the fewest parameters with the smallest use of computational time and memory. Inside the group of evolutionary meta-heuristics, dominated by evolutionary algorithms, there is a simple and effective evolutionary method of optimisation proposed by Boettcher and Percus [1] called extremal optimisation (EO) which is appropriate for solving combinatorial problems. The principle behind EO is to eliminate the weakest or the least adapted component value and replace it by a random one at each iteration without having to tune many parameters.

A hybrid extremal optimisation (HEO) algorithm is presented which finds solutions through the collaborative use of EO with an improved tailor-made local search technique for BPP proposed by Falkenauer [2]. The original technique consists of redistributing a series of items between non-full bins using three different steps. The aim is to reduce the number of necessary bins needed to contain all the items by at least one for each application of the algorithm. In the

proposed local search approach, we add a fourth step that achieves improved results. The EO part of this approach allows the solution state to move between feasible and infeasible spaces. This movement permits the exploration of the search space without being stuck in a particular region of the landscape. The results show that the new hybrid algorithm is able to obtain optimal results for large problems. The advantage of this proposal is its simplicity of implementation compared to other techniques [2, 3, 4, 5, 6, 7].

Given this new approach, we test the behaviour of HEO on the BPP using well-known benchmark problems from the OR-Library [8]. The results show that the proposed algorithm is an effective approach and competitive with other methods [2, 3, 6, 7].

The rest of this paper is organised as follows. In Section 2, a summary of EO is given while Section 3 explains the HEO algorithm to solve the BPP. Section 4 presents an analysis of the obtained results. Finally, in Section 5 we conclude and discuss the future work arising from this research.

2 Extremal Optimisation

EO is an evolutionary meta-heuristic proposed by Bak and Sneppen [9] based on a model of co-evolution between species. This model describes species' evolution via extinction events as a self-organised critical (SOC) [10] process. SOC tries to explain the manifestation of complex phenomena in nature such as the formation of sand piles and the occurrences of earthquakes [11]. The main characteristic is that a power law describes the events in the system. Simply put, these systems have some critical points that are configured in a particular way. When an event converges toward one of these points, a critical state is reached which triggers a series of changes over the elements nearby to the critical point. The system is self-organised to reach a new state of equilibrium. Finally, the system evolves in a transient period of stability until then next critical point is reached.

A good way to understand EO's characteristics is to compare it with another well-known method such as the genetic algorithms (GA) [12]. First, a GA commonly has a set of parameters to be tuned for proper operation; however, in EO only one specific parameter must be tuned. Second, in EO the fitness value is not calculated for each structure that represents a solution as in a GA but for each component of the structure. Each component is evaluated according to its contribution in obtaining the best solution. Third, canonical EO works with a single solution instead of a population of solutions as in GA. Last, EO removes the worst components for the next generations; in contrast, GA promotes a group of elite solutions.

There is only one EO specific parameter that is often referred to as τ [13]. This parameter is used probabilistically to choose the component value to be changed at each iteration of the algorithm. The algorithm ranks the components and assigns to them a number from 1 to n using the fitness of each one (where n in the number of components). Therefore, the fitness must be sorted from the worst to the best evaluated. Then a selection method such as roulette wheel

(RWS) is used to choose the component whose value will be changed to a random one according to the probability calculated for each component as is shown in Equation 1. If the new component makes the solution the best found so far, according to the evaluation function, then this solution is saved as X_{best}. The EO process is shown in Algorithm 1.

$$P_i = i^{-\tau} \qquad \forall i \quad 1 \leq i \leq n, \quad \tau \geq 0 \tag{1}$$

where:
n is the total number of components evaluated and ranked, and
P_i is the probability that the i^{th} component is chosen.

Algorithm 1 Standard EO pseudo-code for minimisation problem

Generate an initial random solution X=$(x_1, x_2, \ldots, x_n)$ and set $X_{\text{best}} = X$;
Generate the probabilities array P according to Equation 1;
for a preset number of iterations **do**
  Evaluate and rank fitness λ_i for each x_i from worst to best, $1 \leq i \leq n$;
  j = Select component based on the probability of its rank P_i using RWS;
  x_j = Generate a random appropriate value that is not equal to x_j;
  $Eva(X)$ = Evaluate the new solution;
  if $Eva(X) < Eva(X_{\text{best}})$ **then** $X_{\text{best}} = X$;
end for
Return X_{best} and $Eva(X_{\text{best}})$;

3 HEO for the BPP

This section shows how the HEO approach is implemented to deal with the BPP. Section 3.1 presents a brief and concise definition of the BPP. Section 3.2 describes the improved local search developed for the BPP. Section 3.3 introduces the HEO approach for the BPP using a integrated version of EO with the proposed local search mechanism.

3.1 The Bin Packing Problem

Many production and distribution tasks require that a series of items with different shapes, sizes, or weights be packed into bins or boxes with a limited capacity. The BPP is a known $\mathcal{NP}$-hard combinatorial optimisation problem (COP) [14] and the one-dimensional bin packing problem is the simplest version of this problem and can be formally described as follows.

A infinite set of bins with a one size bin capacity $C > 0$, and a finite set of n items $w_i = \{w_1, w_2, \ldots, w_n\}$ with different weights among $0 < w_i \leq C$ is to be packed. Find the smallest number m of bins needed to contain all of the n items such that the weight of the items packed in each bin must not exceed the bin capacity C, and that one item w_i can be in one and only one bin.

The bin packing problem could be applied to tasks such as to backup of tapes or disks of equal capacity, to allocate data onto blocks memory with the same size, or to assign processes on a system with identical parallel processors. All these tasks follow the idea of minimising the wastage of resources in the bins.

3.2 A Local Search for the BPP

Local Search (LS) is a mechanism used by some variants of evolutionary heuristics to improve the quality of solutions they receive. LS is commonly used as an iterative process that starts with a feasible solution and then this is improved by performing local modifications. This process is repeated until the current solution can not become better.

Preliminary research by Hendlass and Randall [4] showed the benefits of applying a LS at EO to solve 15 BPP instances ranging in size from 120 items to 500 items. Motivated by this promising result, we apply a improved LS to help solve a large set of 80 BPP instances ranging in size from 120 items to 1000 items. We take as a base the LS mechanism proposed by Falkenauer [2] which in turn was inspired by the work of Martello and Toth [7]. This LS mechanism was also applied to an ant colony approach by Levine and Ducatelle [6].

The augmented improved LS implementation can be explained as follows. The procedure begins by moving all the items contained inside the two least full bins into a (temporary) free bin. Four sequential steps are then applied for each bin that is not full. These steps consist of interchanging one or two items between the current bin and the bin with free items so that the current bin could be refilled up to the limit. The first step swaps two current items by two free items, the second step swaps two current items by one free item and the third step swaps one current item by one free item. The new fourth step swaps one current item by two free items. Next, free items are reinserted into bins provided that the latter have enough space to contain it. Finally, the remaining items in the free bin are put them into a new bin.

This local search process is repeated while new solutions are feasible. The aim of this local search is to reduce by one the number of necessary bins to contain every item.

3.3 A HEO implementation for the BPP

One of the main characteristics of EO is its ability to locate a possible optimal solution stochastically in search space. This characteristic enables EO to avoid being trapped in a local optimum; however, this characteristic also hinders EO from being able to refine the search when a solution is near to the optimal result. Recent research has reported the achievement of better results on some optimisation problems when a local search technique complements EO [4, 15, 16, 17]. Therefore, HEO is made up of an EO framework and a local search procedure that is executed every time a feasible solution is found.

HEO starts by generating an initial solution using the Best Fit Decreasing (BFD) strategy [18]. BFD sorts the items in decreasing order according to its

weight and then trys to put an item into a bin that has minimal free space. BFD guarantees to use no more than $\frac{11}{9}B+1$ bins [18], where B is the optimum number of bins. The reason for generating an initial solution using BFD, rather than purely at random, is based on our interest in minimising the initial numbers of bins necessary to contain all the items. That is done with the goal of making the process more efficient.

As solutions in EO can move between infeasible and feasible space (see Randall [17]), the next step is to carry out an appropriate selection process for both cases. These processes follow the EO scheme but with a slight modification.

When the current solution is feasible, the bin with the largest free space available is selected since the aim of BPP is to minimise the numbers of bins. One item is picked randomly from the selected bin to be moved into other available bin which is chosen according to the new EO rules. Hence, these bins are ranked by a fitness value which is defined as follows:

$$\lambda_j = \frac{R_j}{N_j} \qquad \forall j \text{ such that } 0 < R_j < C \tag{2}$$

where:
R_j is the current weight of the j^{th} bin,
N_j is the number of items in the j^{th} bin, and
λ_j is the fitness of the j^{th} bin.

The idea of this selection process is to try to put the item in a bin where the relation $used\_space/number\_of\_items$ is the smallest. This means that between two bins with the same free space to receive a new item, it is preferable to use the bin with more items of small size. This is because if the bin gets overloaded then it is easier to adjust several small items than a few big items within the bin for the infeasible solution procedure and so to obtain a feasible solution again.

On the other hand, when the current solution is infeasible, the most overloaded bin is selected since this bin degrades the solution. A light item is chosen with a greater probability than a heavy item from the selected bin as it is less likely to overload another bin. Next, the bin to which the item will be added is chosen between non-full bins according to the EO rules. Thus, these bins are ranked by a fitness value using Equation 2.

Finally, the new solution is accepted and evaluated to see if it is feasible or not. In the case that a better feasible solution is obtained, the best solution found so far is updated and the local search process is invoked. Algorithm 2 shows the steps to solve the BPP using the HEO approach.

4 Computational Experiments

The proposed HEO method was coded in the C language and compiled with **gcc** version 4.3.0. The computing platform used to perform the tests has a 1.86 GHz Intel Core2 CPU and 917 MB of memory, and runs under Linux.

The set test used in this paper comes from the OR-Library [8]. This test was contributed by Falkenauer [2] and consists of 80 problems which are divided

Algorithm 2 Pseudocode of the HEO model for the BPP

```
Initialise a feasible solution S using the BFD method and set S_best ← S ;
Generate the probabilities array P according to Equation 1;
S_new = LocalSearch(S);
Evaluate the new solution Eva(S_new);
if Eva(S_new) < Eva(S_best) then S_best ← S_new;
for a preset number of iterations do
  if the current solution is feasible then
    Select the emptiest bin and choose a random item;
  else
    Select the most overloaded bin and choose a light item;
  Evaluate and rank the fitness λ_i for the remaining bins with free space using Eq. 2;
  Choose a bin based on the probability of its rank P_i using RWS;
  Put the selected item into the chosen bin;
  Evaluate the new solution Eva(S_new);
  if S_new is feasible then
    if Eva(S_new) < Eva(S_best) then S_best ← S_new;
    S_new = LocalSearch(S_new);
    if Eva(S_new) < Eva(S_best) then S_best ← S_new;
end for
```

into 4 groups of 20 problems. Each of these has 120, 250, 500 and 1000 items. The items' weight are spread uniformly between 20 and 100 and the capacity for all bins is 150. The problem's name is presented through the nomenclature uNNN_PP where NNN is the number of items and PP is the problem's identification. Each problem is accompanied by its theoretical optimal result.

All results are presented as a percentage gap determined using the theoretical optimal solution and the solution obtained by each analysed method. It is defined as $\%gap = \frac{b-a}{b} \times 100$, where a is the cost of the obtained solution and b is the cost of the theoretical optimal solution. A value of 0 means that the method found the optimal.

The first part of the experiments describes the results when a local search mechanism is applied. Here, the local search method proposed by Falkenauer [2] is compared with the improved version proposed in this paper when one new step is added. Table 1 shows the results of the BSD method, the 3 step Falkenauer method and the 4 step proposed method. The latter two were tested using the BSD result as the initial solution. As can be seen, the improvement made in the new version with respect to the Falkenauer's local search method gave better results for 52 out of 74 problems where the initial solution is not optimal. Also, the average percentage gap for the four groups decreased from around 1.1% to under 0.67%. That means that for the four groups the improved local search method was able to find closer results to the theoretical solution. Thus, the improved local search method is chosen to be used in the next part when the HEO approach is implemented.

In the second part, HEO is configured to run 10 test trials. The number of iterations to complete a HEO process is of 100000. The τ parameter is set at

Problem name	Theo Opt	BFD	% gap	FLS	% gap	PLS	% gap
U120_00	48	49	2.08	49	2.08	48	**0**
U120_01	49	49	**0**	49	**0**	49	**0**
U120_02	46	47	2.17	46	**0**	46	**0**
U120_03	49	50	2.04	50	2.04	49	**0**
U120_04	50	50	**0**	50	**0**	50	**0**
U120_05	48	49	2.08	48	**0**	48	**0**
U120_06	48	49	2.08	49	2.08	49	2.08
U120_07	49	50	2.04	50	2.04	49	**0**
U120_08	50	51	2	51	2	51	2
U120_09	46	47	2.17	47	2.17	47	2.17
U120_10	52	52	**0**	52	**0**	52	**0**
U120_11	49	50	2.04	50	2.04	49	**0**
U120_12	48	49	2.08	49	2.08	49	2.08
U120_13	49	49	**0**	49	**0**	49	**0**
U120_14	50	50	**0**	50	**0**	50	**0**
U120_15	48	49	2.08	49	2.08	48	**0**
U120_16	52	52	**0**	52	**0**	52	**0**
U120_17	52	53	1.92	53	1.92	53	1.92
U120_18	49	50	2.04	49	**0**	49	**0**
U120_19	49	50	2.04	50	2.04	50	2.04
Average			1.44		1.13		**0.62**

Problem name	Theo Opt	BFD	% gap	FLS	% gap	PLS	% gap
U500_00	198	201	1.52	201	1.52	200	1.01
U500_01	201	204	1.49	203	1	202	0.5
U500_02	202	205	1.49	204	0.99	203	0.5
U500_03	204	207	1.47	207	1.47	206	0.98
U500_04	206	209	1.46	209	1.46	208	0.97
U500_05	206	207	0.49	207	0.49	206	**0**
U500_06	207	210	1.45	210	1.45	209	0.97
U500_07	204	207	1.47	207	1.47	206	0.98
U500_08	196	199	1.53	198	1.02	198	1.02
U500_09	202	204	0.99	203	0.5	202	**0**
U500_10	200	202	1	202	1	201	0.5
U500_11	200	203	1.5	203	1.5	202	1
U500_12	199	202	1.51	202	1.51	201	1.01
U500_13	196	198	1.02	198	1.02	196	**0**
U500_14	204	206	0.98	206	0.98	204	**0**
U500_15	201	204	1.49	203	1	202	0.5
U500_16	202	205	1.49	204	0.99	203	0.5
U500_17	198	201	1.52	201	1.52	200	1.01
U500_18	202	205	1.49	204	0.99	203	0.5
U500_19	196	199	1.53	198	1.02	199	1.53
Average			1.34		1.14		**0.67**

Problem name	Theo Opt	BFD	% gap	FLS	% gap	PLS	% gap
U250_00	99	100	1.01	100	1.01	99	**0**
U250_01	100	101	1	101	1	100	**0**
U250_02	102	104	1.96	103	0.98	103	0.98
U250_03	100	101	1	101	1	100	**0**
U250_04	101	102	0.99	102	0.99	102	0.99
U250_05	101	104	2.97	103	1.98	102	0.99
U250_06	102	103	0.98	103	0.98	102	**0**
U250_07	103	105	1.94	104	0.97	104	0.97
U250_08	105	107	1.9	106	0.95	106	0.95
U250_09	101	102	0.99	102	0.99	102	0.99
U250_10	105	106	0.95	106	0.95	106	0.95
U250_11	101	103	1.98	102	0.99	102	0.99
U250_12	105	107	1.9	107	1.9	106	0.95
U250_13	102	104	1.96	104	1.96	103	0.98
U250_14	100	101	1	101	1	100	**0**
U250_15	105	107	1.9	107	1.9	106	0.95
U250_16	97	99	2.06	98	1.03	98	1.03
U250_17	100	101	1	101	1	100	**0**
U250_18	100	102	2	102	2	101	1
U250_19	102	103	0.98	103	0.98	102	**0**
Average			1.52		1.23		**0.64**

Problem name	Theo Opt	BFD	% gap	FLS	% gap	PLS	% gap
U1000_00	399	403	1	403	1	401	0.5
U1000_01	406	411	1.23	410	0.99	407	0.25
U1000_02	411	416	1.22	414	0.73	415	0.97
U1000_03	411	416	1.22	416	1.22	415	0.97
U1000_04	397	402	1.26	400	0.76	399	0.5
U1000_05	399	404	1.25	404	1.25	403	1
U1000_06	395	399	1.01	399	1.01	396	0.25
U1000_07	404	408	0.99	408	0.99	406	0.5
U1000_08	399	404	1.25	402	0.75	403	1
U1000_09	397	404	1.76	402	1.26	403	1.51
U1000_10	400	404	1	404	1	401	0.25
U1000_11	401	405	1	405	1	403	0.5
U1000_12	393	398	1.27	397	1.02	395	0.51
U1000_13	396	401	1.26	401	1.26	397	0.25
U1000_14	394	400	1.52	399	1.27	397	0.76
U1000_15	402	408	1.49	407	1.24	406	1
U1000_16	404	407	0.74	407	0.74	405	0.25
U1000_17	404	409	1.24	408	0.99	407	0.74
U1000_18	399	403	1	402	0.75	402	0.75
U1000_19	400	406	1.5	406	1.5	403	0.75
Average			1.21		1.04		**0.66**

Table 1. Results for the initial BSD solution, Falkenauer local search (FLS) and the proposed local search (PLS). Note that the number of bins and the % gap are shown.

1.4 which has been reported in previous research [4, 17, 19] as a good value to obtain efficient solutions.

The results generated by the HEO approach, shown in Table 2, are compared with those reported by Martello and Toth (MTP) [7], Alvim et al. (LS-BPP) [3], Falkenauer (HGGA) [2], Levine and Ducatelle (HACO) [6], and Randall, Hendtlass and Lewis (SEO, PEO) [3] [20]. For the four methods that do not use EO, the results are presented only using one column that shows the % gap with respect to the theoretical solution. The remaining three methods that use EO, the results are presented by three columns, "mi%", "me%", and "ma%" which denote the minimum, median and maximum % gap respectively.

[3] SEO means EO with a single solution. POE means EO with a population of solutions.

Problem	Theo	MTP	LS-BPP	HGGA	HACO	SEO			PEO			HEO		
name	Opt	%	%	%	%	mi%	me%	ma%	mi%	me%	ma%	mi%	me%	ma%
U120_00	48	0	0	0	0	0	0	2.08	0	0	0	0	0	0
U120_01	49	0	0	0	0	0	0	0	0	0	0	0	0	0
U120_02	46	0	0	0	0	0	0	0	0	0	0	0	0	0
U120_03	49	0	0	0	0	0	2.04	4.08	1.02	2.04	2.04	0	0	0
U120_04	50	0	0	0	0	0	0	0	0	0	0	0	0	0
U120_05	48	0	0	0	0	0	0	0	0	0	2.08	0	0	0
U120_06	48	0	0	0	0	0	0	2.08	0	0	2.08	0	0	0
U120_07	49	0	0	0	0	0	0	0	0	0	0	0	0	0
U120_08	50	2	2	2	0	0	0	1.96	0	0	0	0	0	0
U120_09	46	0	2.17	0	0	0	1.09	2.17	0	2.17	2.17	0	0	0
U120_10	52	0	0	0	0	0	0	0	0	0	0	0	0	0
U120_11	49	0	0	0	0	0	0	2.04	0	0	0	0	0	0
U120_12	48	0	0	0	0	0	2.08	2.08	0	2.08	2.08	0	0	0
U120_13	49	0	0	0	0	0	0	0	0	0	0	0	0	0
U120_14	50	0	0	0	0	0	0	0	0	0	0	0	0	0
U120_15	48	0	0	0	0	0	0	2.08	0	0	0	0	0	0
U120_16	52	0	0	0	0	0	0	1.92	0	0	0	0	0	0
U120_17	52	0	0	0	0	0	0	5.77	0	1.92	1.92	0	0	0
U120_18	49	0	0	0	0	0	0	0	0	0	0	0	0	0
U120_19	49	2.04	2.04	2.04	0	0	0	2	0	0	0	0	0	0
Average		0.2	0.31	0.2	0	0	0.26	1.41	0.05	0.41	0.62	0	0	0

Problem	Theo	MTP	LS-BPP	HGGA	HACO	SEO			PEO			HEO		
name	Opt	%	%	%	%	mi%	me%	ma%	mi%	me%	ma%	mi%	me%	ma%
U250_00	99	1.01	0	0	0	0	0.51	1.01	0	1.01	1.01	0	0	0
U250_01	100	0	0	0	0	0	0	1	0	0	1	0	0	0
U250_02	102	0	0.98	0	0	0	0.49	0.98	0	0.98	0.98	0	0	0
U250_03	100	0	0	0	0	0	0	0	0	0	1	0	0	0
U250_04	101	0	0	0	0	0	0	0.99	0	0.99	0.99	0	0	0
U250_05	101	1.98	0.99	0	0	0	0.99	1.98	0.99	0.99	1.98	0	0	0.99
U250_06	102	0	0	0	0	0	0	0	0	0	0	0	0	0
U250_07	103	0.97	0.97	0.97	0	0	0	3.85	0	0	0	0.97	0.97	0.97
U250_08	105	0.95	0.95	0	0	0.95	0.95	0.95	0.95	0.95	1.9	0	0	0
U250_09	101	0.99	0	0	0	0	0.99	2.97	0.99	0.99	0.99	0	0	0
U250_10	105	0.95	0	0	0	0	0	1.9	0	0.95	0.95	0	0	0
U250_11	101	0.99	0.99	0	0	0.99	0.99	0.99	0.99	0.99	0.99	0	0	0
U250_12	105	0.95	0.95	0.95	0.95	0	0	0.94	0	0	0.94	0.95	0.95	0.95
U250_13	102	0.98	0.98	0.98	0.98	0	0	0.97	0	0	0.97	0.98	0.98	0.98
U250_14	100	0	0	0	0	0	0	1	0	1	1	0	0	0
U250_15	105	0.95	0.95	0	0	0.95	0.95	3.81	0.95	1.9	1.9	0	0.95	0.95
U250_16	97	1.03	1.03	0	0	0	0	3.09	1.03	1.03	1.03	0	0	0
U250_17	100	0	0	0	0	0	0	1	0	0	0	0	0	0
U250_18	100	0	0	0	0	1	1	1	1	1	1	0	0	0
U250_19	102	0	0	0	0	0	0	0.98	0	0	0.98	0	0	0
Average		0.59	0.44	0.15	0.1	0.19	0.34	1.47	0.35	0.64	0.98	0.15	0.19	0.24

Problem	Theo	MTP	LS-BPP	HGGA	HACO	SEO			PEO			HEO		
name	Opt	%	%	%	%	mi%	me%	ma%	mi%	me%	ma%	mi%	me%	ma%
U500_00	198	1.52	0.51	0	0	0.51	0.51	2.53	0.51	1.01	1.01	0	0	0
U500_01	201	0.5	0.5	0	0	0.5	0.5	0.5	0.5	1	1	0	0	0.5
U500_02	202	0.99	0	0	0	0	0.5	0.99	0.5	0.74	0.99	0	0	0
U500_03	204	0.98	0.49	0	0	0.49	0.49	1.47	0.98	0.98	0.98	0	0	0.49
U500_04	206	1.46	0	0	0	0	0.49	1.94	0.49	0.49	0.97	0	0	0
U500_05	206	0.49	0	0	0	0	0.97	2.91	0.49	0.73	0.97	0	0	0
U500_06	207	1.45	0.48	0	0	0.48	0.72	1.45	0.97	0.97	1.45	0	0	0.48
U500_07	204	1.47	0.49	0	0	0.49	1.23	3.43	0.98	1.47	1.96	0	0.25	0.49
U500_08	196	1.02	0.51	2	0	0	0.26	1.02	0.51	0.51	1.02	0	0	0.51
U500_09	202	0.99	0	0	0	0	0	0.5	0	0.5	0.99	0	0	0
U500_10	200	1	0.5	0	0	0	0.5	0.5	0.5	0.5	1	0	0	0
U500_11	200	1	0.5	0	0	0.5	0.5	3	0.5	1	1.5	0	0	0.5
U500_12	199	1.51	0.5	0	0	0.5	0.5	1.01	0.5	1.01	1.01	0	0	0.5
U500_13	196	0.51	0	0	0	0	0.51	1.53	0	0.51	0.51	0	0	0
U500_14	204	0.49	0	0	0	0.49	0.49	8.33	0.49	0.49	0.98	0	0	0
U500_15	201	1	0	0	0	0.5	0.5	0.5	0.5	0.5	1	0	0	0
U500_16	202	0.99	0	0	0	0	0	0.99	0	0.5	0.5	0	0	0
U500_17	198	1.52	0	0	0	0.51	0.51	0.51	0.51	1.01	1.01	0	0	0.51
U500_18	202	1.49	0	0	0	0	0.5	1.98	0.99	1.49	1.49	0	0	0
U500_19	196	1.53	0.51	2.04	0	0.51	0.51	1.02	0.51	1.02	1.53	0	0	0
Average		1.09	0.25	0.2	0	0.27	0.51	1.81	0.52	0.82	1.09	0	0.01	0.2

Problem	Theo	MTP	LS-BPP	HGGA	HACO	SEO			PEO			HEO		
name	Opt	%	%	%	%	mi%	me%	ma%	mi%	me%	ma%	mi%	me%	ma%
U1000_00	399	1	0	0	0							0	0	0.25
U1000_01	406	0.99	0	0	0							0	0	0.25
U1000_02	411	1.22	0	0	0							0	0	0
U1000_03	411	1.22	0.49	0	0							0	0.24	0.49
U1000_04	397	1.01	0.25	0	0							0	0.25	0.5
U1000_05	399	0.75	0.25	0	0							0	0	0.25
U1000_06	395	0.76	0	0	0							0	0	0
U1000_07	404	0.5	0	0	0							0	0	0.25
U1000_08	399	0.75	0	0	0							0	0	0.25
U1000_09	397	1.26	0.5	0	0							0	0	0.25
U1000_10	400	1	0	0	0							0	0	0
U1000_11	401	0.75	0.25	0	0							0	0	0.25
U1000_12	393	0.76	0	0	0							0	0	0.25
U1000_13	396	1.26	0	0	0							0	0.25	0.25
U1000_14	394	1.27	0.51	0	0							0	0.13	0.25
U1000_15	402	1.24	0.25	0	0							0	0	0.25
U1000_16	404	0.74	0	0	0							0	0	0.25
U1000_17	404	0.74	0.25	0	0							0	0	0.25
U1000_18	399	1	0	0	0							0	0	0
U1000_19	400	1.25	0	0	0							0	0	0.25
Average		0.97	0.14	0	0							0	0.04	0.22

Table 2. Comparative HEO test results. Note that blank spaces are present because these problems were not solved by the authors.

HEO is confirmed to be an efficient and competitive approach to solve the BPP. In 77 out of 80 problems, the theoretical optimal result was found, only one less than HACO which found 78. The result of the three remaining bins (u250_07, u250_12, u250_13) is only one bin from the theoretical optimal solution. In fact, the problem u250_13 has no solution for the theoretical optimal as is reported in Levine and Ducatelle [6].

Despite the good results obtained by the previous two works where EO is applied (SEO and PEO), we can infer from the development of HEO that the better results are due to the use of a good initial solution, the improved local search mechanism and the modified EO selection process.

In Table 3 we can observe the excellent runtime of HEO in relation to the other methods. Indeed, the difference in the average time among four test group is quite similar which shows a balanced performance in relation to the number of items to be packed.

Problem name	MTP time	LS-BPP time	HGGA time	HACO time	HEO time
u120	370	0.2	381	1	1
u250	1516	6.7	1337	52	1.2
u500	1535	37.25	1015	50	1.2
u1000	9393	143.55	7059	147	1.8

Table 3. Average time, in seconds, for each group test.

5 Conclusions

This paper has described a Hybrid EO approach for the BPP. EO is a recent form of search for which there exists a relatively small amount of work that is applicable to COPs. Some of the research has been in combining EO with a local search mechanism to obtain better solutions. The strength of HEO, which is EO and improved local search, lies in exploiting the stochastic nature of EO as a coarse grain solver to move solutions from stagnant positions to explore new regions toward the optimal result. Besides, the improved local search mechanism works in a fine grain way to find better solutions from the last found feasible solution.

Results obtained for the uniformly distributed test set with HEO have been encouraging. This approach has been able to find the optimal solution for 77 out of 80 problems. For the three remaining problems, the result was only 1 item from the theoretical optimal solution. Note that for one of these three problems, it is not possible to find the optimal solution.

When the proposed HEO is compared to the existing evolutionary approaches, we can see that HEO is able to obtain quality solutions as good as the best of them. The advantage that our approach has low requirements of runtime, memory, parametrisation and implementation.

The proposed approach could be adapted to solve other problems such as the cutting stock problem, the multiprocessor scheduling problem, and the graph colouring problem. Also, we are looking to adapt the algorithm presented to solve the multi-objective version of BPP.

References

[1] Boettcher, S., Percus, A.G.: Evolutionary strategies extremal optimization: Methods derived from co-evolution. In: GECCO-99: Proceedings of the Genetic and Evolutionary Computation Conference. (1999) 825–832

[2] Falkenauer, E.: A hybrid grouping genetic algorithm for bin packing. Journal of Heuristics **2**(1) (1996) 5–30

[3] Alvim, A., Glover, F., Ribeiro, C., Aloise, D.: Local search for the bin packing problem. In: Extended Abstracts of the III Metaheuristics International Conference (MIC99), Angra dos Reis, Brazil (1999) 7–12

[4] Hendtlass, T., Randall, M.: Extremal optimisation and bin packing. In: SEAL. Volume 5361 of Lecture Notes in Computer Science., Springer (2008) 220–228

[5] Karmarkar, N., Karp, R.M.: The differencing method of set partitioning. Technical Report UCB/CSD-83-113, EECS Department, University of California, Berkeley, Berkeley, CA, USA (1983)

[6] Levine, J., Ducatelle, F.: Ant colony optimisation and local search for bin packing and cutting stock problems. Journal of the Operational Research Society **55**(7) (2004) 705–716

[7] Martello, S., Toth, P.: Lower bounds and reduction procedures for the bin packing problem. Discrete Applied Mathematics **28**(1) (1990) 59–70

[8] Beasley, J.E.: OR-Library: Distributing test problems by electronic mail. Journal of the Operational Research Society **41**(11) (1990) 1069–1072

[9] Bak, P., Sneppen, K.: Punctuated equilibrium and criticality in a simple model of evolution. Physical Review Letters **71**(24) (1993) 4083–4086

[10] Bak, P., Tang, C., Wiesenfeld, K.: Self-organized criticality: An explanation of the 1/f noise. Physical Review Letters **59**(4) (1987) 381–384

[11] Bak, P.: How nature works. Springer-Verlag New York Inc. (1996)

[12] Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1989)

[13] Boettcher, S.: Extremal optimization: Heuristics via co-evolutionary avalanches. Computing in Science and Engineering **2** (2000) 75–82

[14] Garey, M.R., Johnson, D.S.: Computers and Intractability : A Guide to the Theory of NP-Completeness. Series of Books in the Mathematical Sciences. W. H. Freeman & Co., New York, NY, USA (1979)

[15] Huang, W., Liu, J.: Extremal optimization with local search for the circular packing problem. In: Proceedings of the Third International Conference on Natural Computation, ICNC '07. Volume 5., IEEE Computer Society (2007) 19–23

[16] Gómez-Meneses, P., Randall, M.: Extremal optimisation with a penalty approach for the multidimensional knapsack problem. In: SEAL. Volume 5361 of Lecture Notes in Computer Science., Springer (2008) 229–238

[17] Randall, M.: Enhancements to extremal optimisation for generalised assignment. In: ACAL. Volume 4828 of Lecture Notes in Artificial Intelligence., Springer (November 2007) 369–380

[18] Martello, S., Toth, P.: Knapsack problems: algorithms and computer implementations. John Wiley & Sons, Inc., New York, NY, USA (1990)

[19] Boettcher, S., Percus, A.G.: Extremal optimization for graph partitioning. Physical Review E **64** (2001) 026114

[20] Randall, M., Hendtlass, T., Lewis, A.: Extremal optimisation for assignment type problems. In: Biologically-Inspired Optimisation Methods: Parallel Algorithms, Systems and Applications. Volume 210 of Studies in Computational Intelligence. Springer-Verlag (2009) 139–164