Received: May 11, 2020. Revised: July 22, 2020.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Comparing Various Genetic Algorithm Approaches for Multiple-choice Multi-dimensional Knapsack Problem (mm-KP)

Admi Syarif[1]* Dian Anggraini[1] Kurnia Muludi[1]
Wamiliana[2] Mitsuo Gen[3]

[1]Department of Computer Science, Faculty of Mathematics and Sciences, Lampung University Jl. S. Brodjonegoro No.1, Bandar Lampung, Indonesia, 35145
[2]Department of Mathematics Faculty of Mathematics and Sciences, Lampung University Jl. S. Brodjonegoro No.1, Bandar Lampung, Indonesia, 35145
[3]Deparment of Computer Science, Tokyo University of Science, Tokyo, Japan
* Corresponding author's admi.syarif@fmipa.unila.ac.id

Abstract: One of the essential and well-known classes of combinatorial optimization problems commonly studies by researchers in the last five decades is called the Knapsack Problem (KP). Many variants of KP have been introduced for different real-world applications. Among them, the multiple-choice multi-dimensional Knapsack Problem (mm-KP) is the most complex model with an NP-hard problem. Several authors have reported the robustness of heuristics for mm-KP; irrespective of its advantages, no method currently has the ability to solve the problem optimally all time. This paper aims to determine the best GA strategy and evaluate the performance of several heuristic algorithms to solve mm-KP. We investigate the use of two techniques that are included in the GA approach. The first, two different strategies are adopted to handle infeasible chromosomes, namely penalty and repairing procedure. Second, we develop a new-simple local search to improve the quality of the solution found. Experimental studies on the 13 (thirteen) Benchmark instances are conducted to evaluate the effectiveness of the approach based on solutions quality, the number of the optimal solution reached, and average errors. The results showed that hr-GA tends to reach optimal/near-optimal solutions. Furthermore, the results from studies on heuristic algorithms also show that hr-GA is a promising approach, with local search used to immensely improve the quality.

Keywords: Repairing strategy, Local search, Heuristic method, Genetic algorithm, Knapsack problem.

1. Introduction

In 1950, Dantzig introduced the classical Knapsack Problem (KP), which is one of the famous combinatorial optimization problems. It represents the problem of selecting the subsets of the *n* items to maximize the corresponding profit, and the total weight does not exceed the Knapsack capacity. Practical applications of KP can be found in some of our daily life, such as the everyday diet program, an optimal investment plan, cargo loading, cutting stock, budget control, and financial management [1]. Different variants of KP are found in the literature, including, multiple-choice KP (mc-KP), multi-dimensional KP (md-KP), and multiple-choices multi-dimensional KP (mm-KP) [2]. Among them, mm-KP is the most complex belongs to the class of an NP-hard problem. It is considered as a variant of the md-KP where items are divided into groups, and precisely one item per group must be selected. The applications of mm-KP can be found in many real-life applications, including Chip Multiprocessor Run-time Resource Management [3], Global Routing of Wiring in Circuits [4], and Service Level Agreement [5].

The methods proposed in the literature to solve mm-KP can be grouped into two classes: exact and heuristic methods. Since mm-KP is an NP-hard problem, its search space exponentially grows as the problem size increases. Therefore, many researchers pay more attention to develop heuristic methods to

solve mm-KP. Although the heuristics do not guarantee the finding of an optimal solution, those have been reported useful in determining optimal/near-optimal solutions for many hard optimization problems, including mm-KP.

To our knowledge, the first heuristic algorithm to solve mm-KP was reported by Moser et al. [6]. The authors introduced a Lagrangian Relaxation algorithm that was repeatedly permuting to reduce the infeasibility of solutions. A heuristic algorithm based on aggressive resource usage was proposed by Khan et al. [7]. This heuristic algorithm performs better than Moser's Algorithm. Then, Hifi presented several approaches to solve mm-KP. First, Hifi et al. proposed a heuristic approach with a guided-local search that used the principle based on trying several diversified solutions obtained after penalizing the costs of the objective function with penalties parameters [8]. Hifi et al. also developed an algorithm based on a reactive local search to try a diversification search and to escape local optima [9]. The authors reported that their methods are able to outperform Moser's and Khan's algorithms.

Later, Cherfi and Hifi presented column generation methods hybridized with branch-and-bound [10]. They reported that the approaches could obtain better solutions than former approaches on the benchmark instances. An ant colony algorithm approach to solve mm-KP was given by Iqbal et al. [11]. A heuristic method called oscillation (OSC Algorithm) was introduced by Htiouech et al. [12]. The authors used the constraint normalization method to improve the quality of solutions. Xia et al. developed another similar approach to the OSC algorithm, called Stochastic Local Search (SLS) [13]. The algorithm adopted a simple additive weighting scheme to adjust the weight (multiplier) on dimensions.

In one latest paper, Htiouech and Alzaidi proposed a heuristic algorithm called AMMKP [14]. The authors presented the way to decompose the mm -KP into many smaller sub-problems, and each subproblem then solved by an agent. The results show that the approach is able to solve several benchmark instances in the literature effectively.

Since Holland introduced it in 1975 [15], the Genetic Algorithm (GA) has been a prevalent heuristic method. Previous studies have shown the robustness of GA for various hard optimization problems, including the logistic problem [16] [17], scheduling problem [18], vehicle routing problem [19], and so on. For some specific cases, however, it often tends to provide local optimum and takes more time to reach optimal solutions. Therefore, it is crucial to carry out studies on designing an effective and efficient GA approach.

The most essential and influential component associated with the implementation of GA is the method used to represent the chromosome. For many optimization problems, it is difficult to express the chromosome in a way capable of fulfilling the constraint functions. This is because the generated chromosome in the population is either feasible or infeasible. Two strategies are usually adopted to overcome the infeasible situation, namely repairing and penalty [20]. Another critical issue in the applications of GA is associated with the strategies used to quit the local-optimal solution. The most common approach is to develop and hybridize GA with a local search technique [21].

This paper aims to determine the best GA strategy to solve mm-KP. To improve the quality of the solution, we developed a new simple local search and hybridized it into the GA loop. The approaches adopt both repairing and penalty strategy, namely as repairing-based (sr-GA), hybrid repairing-based GA (hr-GA), and hybrid penalty-based GA (hp-GA). Furthermore, some comparisons with other heuristic methods are also made based on the solution quality, average error, and the number of instances solved optimally.

To evaluate the approaches, we conducted some numerical experiments on set Benchmark test problems taken from OR-Library. The 13 (thirteen) widely studied instances are used to measure the performances of the algorithms. The experimental results show that hr-GA has the merit of high effectiveness and can obtain competitive results with the other heuristics.

We organize this paper into five sections. The second outlines the mathematical formulation of mm-KP. The brief designs of the proposed algorithms, including the chromosome representation, genetic operation, and local search technique, are presented in the third section. The numerical experiments and the comparison of results to other methods are described in the fourth section. Finally, conclusions are drawn in the last part.

2. Mathematical model and algorithm

Let

i	index of class
j	index of item
k	index of resource
R_k	resource constraint $R = R_1, R_2, R_3, \dots, R_m$
r_{ij}^k	the need for the k^{th} resource for the j^{th} item in the i^{th} class

mm-KP aims to fill one item from each class of Knapsack in order to satisfy the resource capacity

constraints and maximize the total profit values. Formally the mathematical formulation of mm-KP can be written as follows:

$$\max\ Z = \sum_{i=1}^{n}.\sum_{j=1}^{r_i} x_{ij} v_{ij} \quad (1)$$

s.t.

$$\sum_{i=1}^{n}.\sum_{j=1}^{r_i} r_{ij}^{k}\, x_{ij} \le R_k, k \in\{1 \ldots\ldots, m\} \quad (2)$$

$$\sum_{j=1}^{r_i} x_{ij} = 1, i \in \{1, \ldots\ldots n\} \quad (3)$$

$$x_{ij} \in\{0,1), i\epsilon\{1, \ldots\ldots n\}, j\epsilon\{1, \ldots\ldots r_i\} \quad (4)$$

Here, the value of x_{ij} is either 1 or 0, which implies that item j in the i-th class is chosen, or not chosen. The v_{ij} represents the profit value of item j in the i-th class.

3. Design of the algorithms

3.1 Chromosome representation and evaluation

When implementing GA for a specific application, the first step is to determine a way to represent a possible solution to the problems. We have to generate some feasible chromosomes, which is as much as the desired population size. For this research, we represent chromosome by using a string, that its length is equal to the number of classes. For example, the dataset I01 comprises of five groups (i), five items (j), and five resources (k) in each class, with a resource constraint (R_k) of 25. The chromosome representation for this instance I01 can be illustrated in Fig. 1. The data for this instance (I01) is as follows:

In the above chromosome, the index represents the class, while the value item gene represents the item index j in each class i. From the chromosome, the selected items are 4, 5, 3, 2, and 4 in the 1st, 2nd, 3rd, 4th, and 5th class. The decoding mechanism is done by selecting one item for each category. The number of resource consumption (r_{ijk}) of each item is used to check for the resource constraint. Here, the total value of r_{ijk} cannot exceed the (R_k) amount of each class, as shown in Table 2.

Figure. 1 Chromosome representation for instance I01

Table 1. Data for IO1 Instance

1					
7	1	3	1	1	6
17	1	4	9	9	3
25	4	3	9	8	2
35	4	5	8	0	6
36	6	8	3	0	7
2					
9	0	0	4	4	2
10	0	0	1	8	7
10	1	1	6	0	6
39	9	1	2	2	4
44	8	7	0	8	2
3					
15	2	0	5	5	5
19	2	3	2	6	2
20	3	1	6	4	7
44	6	7	5	6	9
50	9	5	9	2	2
4					
5	0	1	3	8	0
25	2	2	7	0	8
32	5	5	6	1	9
37	6	3	6	9	1
37	7	9	7	2	3
5					
24	4	0	7	0	2
30	4	8	9	0	0
32	5	2	7	2	0
43	5	5	9	5	2
44	9	2	2	2	3

Table 2. The decoding of the chromosome

	4	**5**	**3**	**2**	**4**
r_{ij1}	4	8	3	2	5
rij_2	5	7	1	2	5
r_{ij3}	8	0	6	7	9
r_{ij4}	0	8	4	0	5
r_{ij5}	6	2	7	8	1
R_k	23	25	21	19	25

3.2 Crossover and mutation

When implementing GA, two genetic operations are usually used to involve the new solution space, namely as crossover and mutation. The crossover operation is used to produce new offspring by recombining gen between selected parents. When choosing the crossover method, we have to consider the chromosome representation. There are several variants of the crossover methods introduced in the literature, such as one-point crossover, two-point crossover, PMX, WMX, etc.[8]. For this research, we

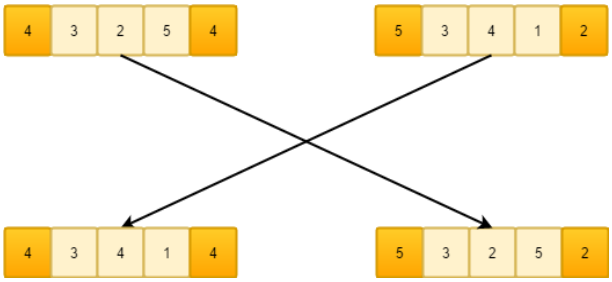

Figure. 2 The illustration of two-point crossover

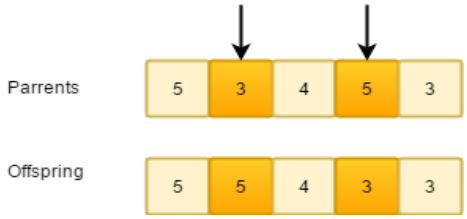

Figure. 3 The illustration of swap mutation operation

adopted the two-point crossover technique, as illustrated in Fig. 2 [22]. The following process explains the steps of the crossover operation.
Step 1: Choose two parents arbitrarily for crossover.
Step 2: Randomly determine two points
Step 3: Exchange the substring between these two points.

3.3 Selection strategy

The last stage of the GA process is the selection strategy to determine the chromosomes for the next generation. This stage adopted the elitist selection method. The *pop_size* best chromosomes (the highest fitness value) are chosen for the next generation.

3.4 Local search

Genetic algorithms (GA) function as a global search technique; however, it may often take a relatively long computational time to converge for a global optimum [24]. To solve this, many researchers suggested hybridizing GA with the local search technique. In this research, we developed a new and simple local search technique called Switching Local Search (SLS). It is done by first selecting a gene in the chromosome and changing its value randomly. The following Algorithm 1 illustrates the overall description of the proposed algorithms.

Algorithm 1: GAs for mm-KP
Input: data for the mm-KP test problem
Output: the best solution

```
Genetic Algorithm {
    Generate Initial population P(t);
    Evaluation Initial population P(t);
  Penalty strategy;
  While (not STOPPING CONDITION)
      Crossover;
      Mutation;
      Evaluation;
      (Penalty/Repairing strategy);
      Selection;
      Local Search Best chromosome P(t);
      P = P(t+l);
    }
}
```

It is shown that two different strategies are used to handle an infeasible chromosome resulted in crossover and mutation operations. The first strategy includes a procedure to repair the infeasible chromosome. The second strategy applies the penalty value for the infeasible chromosome. As mm-KP is the maximization problem, the penalty value will decrease the objective value. Thus, it will reduce the opportunity of the chromosome to be selected for the next generation. The local search procedure was included in the GA loop and implemented to the best chromosome to improve the solution quality.

4. Experimental design and results

4.1 Experimental design

To evaluate the effectiveness of the algorithms, several experimental studies have been conducted on 13 (thirteen) different size Benchmark instances taken from OR-Library. The test problems were divided into 3 (three) groups according to the size; small size (I01-I06), medium (I07-I09), and large (I10-I13). The number of variables in these 13 test problems varies from 25 to 4000. The detail of the

Table 3. Details of the instances, where *n* denotes the number of classes, r_i is the number of items in each group, and *m* is the number of resources

No.	Test problem	Data				Parameter	
		N	r_i	m	R_k	*Pop_size*	*Max_gen*
1	I01	5	5	5	25	50	1000
2	I02	10	5	5	50	50	1500
3	I03	15	10	10	75	50	2000
4	I04	20	10	10	100	100	1000
5	I05	25	10	10	125	100	1500
6	I06	30	10	10	150	100	2000
7	I07	100	10	10	500	200	1000
8	I08	150	10	10	750	200	1500
9	I09	200	10	10	1000	250	2000
10	I10	250	10	10	1250	300	1000
11	I11	300	10	10	1500	300	1500
12	I12	350	10	10	1750	450	2000
13	I13	400	10	10	2000	500	2000

Table 4. Performance of the GA approaches on all instances (SD: Standard Deviation)

Dataset	Optimum	sr-GA			hp-GA			hr-GA		
		Best	Average	SD	Best	Average	SD	Best	Average	SD
I01	173	**173**	173	0	**173**	170	2.9	**173**	173	0
I02	364	**364**	351	6.7	355	346.1	5.2	**364**	361	0.9
I03	1602	1536	1502.5	15.9	1556	1518	14.5	**1600**	1552.5	28.7
I04	3597	3433	3380.8	32.5	3488	3410	39.2	**3571**	3541.4	22
I05	3905.7	3900.4	3800.8	103	3905.7	3892	26.9	**3905.7**	3902	2.6
I06	4799.3	4787.2	4698.3	72.6	4799.3	4769.9	30.5	**4799.3**	4796	4.2
I07	24587	23071	22997.8	39.7	23717	23627	51.1	**23870**	23541.1	79.2
I08	36877	35536	34819.8	345.1	35547	32933	7764	**35626**	34932.4	90.2
I09	49167	47309	46237.6	540.7	47415	45946	748.1	**47370**	47338.1	43.8
I10	61437	58876	57826	568	59226	58898	245.1	**59228**	59056	40.3
I11	73773	70706	70557.3	142.1	70724	66620	1724.7	**71021**	70782.5	99.4
I12	86071	82089	89510.1	212	82305	77908	1195.6	**82627**	82342.8	104.5
I13	98429	94006	93478	279.2	94116	88429	1319.2	**94570**	94321.2	163

sr-GA: standard repairing-based GA (without local search)
hr-GA: hybrid repairing-based GA
hp-GA: hybrid penalty-based GA

test problems used for the experiments is shown in Table 3.

4.2 Results and discussion

A total of three GA approaches were developed, namely standard repairing GA (sr-GA), hybrid penalty-based (hp-GA), and hybridized penalty-based GA (hp-GA) by using Mathlab R2015b and run on PC Processor intel® Core ™i5-3470S. For the experiments, we set the values of crossover probability and mutation probability as 0.4 and 0.2, respectively. Each algorithm is run 20 (twenty) times for each test problem.

Table 4 summarizes the overall results of the experiments, where the best and the average values represent the best and the average fitness value from the 20 (twenty) running times. Standard deviation (SD) is the distribution of statistical measures of the data distribution. Optimum is the best-obtained solution in the literature. The results highlight that repairing-based GA has better performance than penalty-based GA. Due to the difficulties in generating feasible chromosomes, the penalty strategy cannot give good results for all of the test problems. In addition, many offspring, which led to GA operations is also infeasible. It can be seen from these results that combining GA with the local search will improve the solution quality immensely. The overall results show that hr-GA has better performance to solve mm-KP all of the time.

To assess the merit and limit of the proposed approach, we compare the performance of the new hr-GA to other heuristic algorithms available in the following literature: Moser [6], Heuristic [25], Modified-Reactive-Local-Search (MRLS) [9], Khan-Li-Manning-Akbar Algorithm (KLMA) [7], Derived algorithm [8], Ant Colony Algorithm [11], OSC Algorithm [12], SLS Algorithm [13] and AMMKP algorithm [14].

Table 5 summarizes the comparison of the results obtained by the heuristic algorithms. Similarly, in this Table, we also indicate our achievements that have the values greater than or equal to the best-published results in bold. These results show that hr-GA tends to reach optimal/near-optimal solutions to the problem. It can obtain the number of solution that better than or equal to the best solution in 5 cases among 13 cases, which better or same as the results given by Moser (1 case), Heuristic (1 case), RLS (4 cases), KLMA (2 cases), Derived Algorithm (2 cases), Ant Colony algorithm (5 cases), OSC algorithm (4 cases), SLS algorithm (6 cases) and AMMKP (11 cases).

In this research, we calculate the percentage error value for each instance by using the following formula:

$$Error = \frac{best-optimum}{optimum} x\ 100\% \quad (5)$$

The error comparisons of the heuristic methods for each test problem are illustrated in Figure 4. It shows that hr-GA can obtain good quality and reach optimal/optimal solution to the problem most of the time.

In this research, we also analyze the algorithm based on the average errors given by the methods for

Table 5. Results of some heuristic methods on all mm-KP instances

Dataset	Optimum	Moser	Heuristic	RLS	KLMA	Der_ Algo	Ant	OSC	SLS	AMMKP	hr-GA
I01	173	151	167	**173**	167	**173**	**173**	**173**	**173**	**173**	**173**
I02	364	291	354	**364**	354	356	**364**	**364**	**364**	**364**	**364**
I03	1602	1464	1533	1595	1533	1553	**1602**	1594	**1602**	1594	1600
I04	3597	3375	3437	3564	3437	3502	3569	3514	**3592**	**3592**	3571
I05	3905.7	**3905.7**	3899.1	**3905.7**	**3905.7**	3905.2	**3905.7**	**3905.7**	**3905.7**	**3905.7**	**3905.7**
I06	4799.3	4115.2	**4799.3**	**4799.3**	**4799.3**	**4799.3**	**4799.3**	**4799.3**	**4799.3**	**4799.3**	**4799.3**
I07	24857	23556	23912	24121	23912	23983	24159	24162	24311	**24310**	24170
I08	36877	35373	35979	36110	35979	36007	36240	36405	36463	36530	**36641**
I09	49167	47205	47901	48291	47901	48048	48367	48567	48580	**48711**	48191
I10	61437	58648	59811	60291	59811	60176	60475	60858	60661	**60911**	60228
I11	73773	70532	71760	72283	71760	72003	72558	73022	72778	**73200**	72003
I12	86071	82377	84141	84446	84141	84160	84707	85284	84889	**85338**	85015
I13	98429	94166	96003	96850	96003	96103	96834	97545	97082	**97744**	97050
#Best		**1**	**1**	**4**	**2**	**2**	**5**	**4**	**6**	**11**	**5**

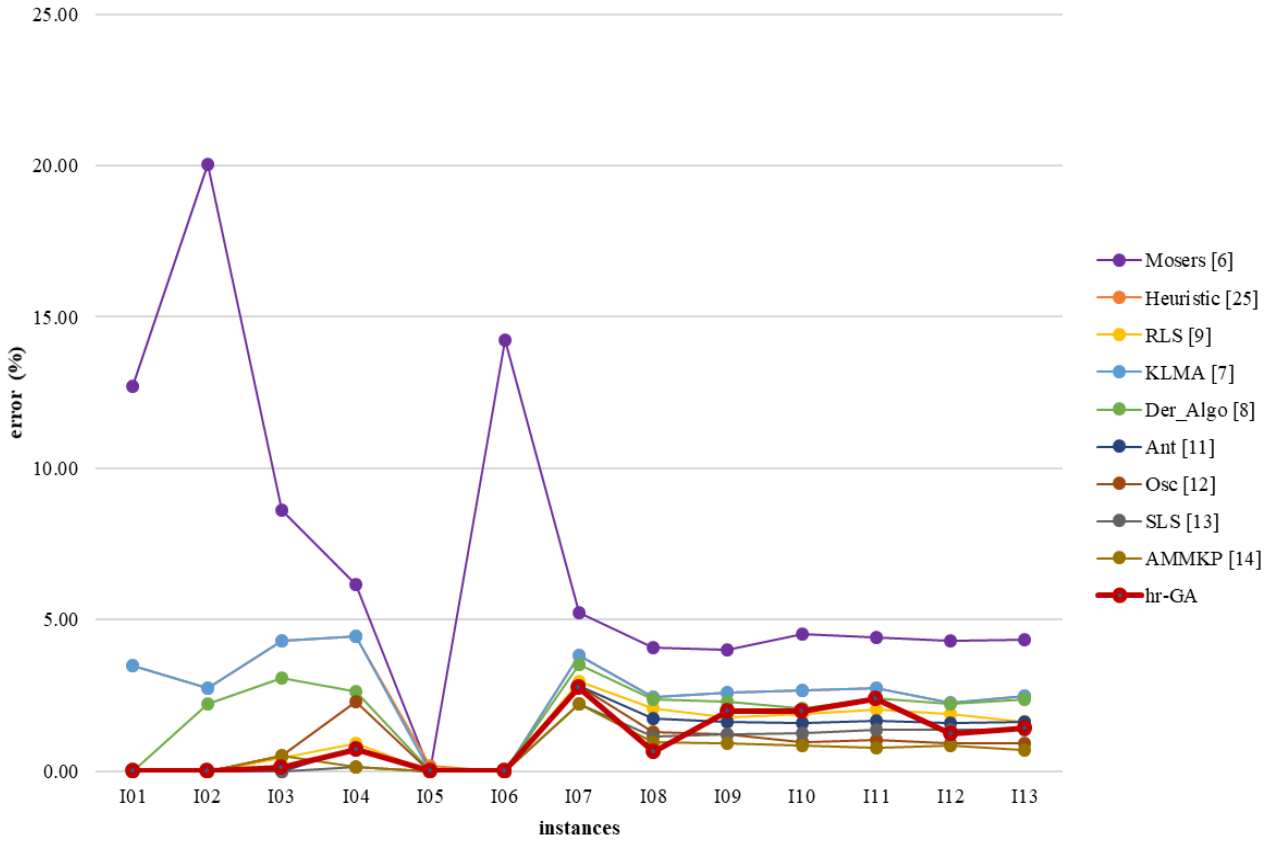

Figure 4. Error comparison between heuristic approaches on all mm-KP instances

those 13 test problems. The computation indicates that hr-GA provides competitive solutions with the average error equal to 1.02%, better than those obtained by Moser (7.13%), Heuristic (2.62%), RLS (1.2%), KLMA (2.6%), and Derived algorithm (1.93%), and the ant colony algorithm (1.03%). However, in comparison to the OSC algorithm (0.91%), SLS algorithm (0.77%), and AMMKP (0.61%), it still has a slightly larger average error. We illustrate the comparison of the average error for each method in the following Figure 5.

5. Conclusion

This paper analyzed three different GA approaches to solving mm-KP. The proposed methods adopt different strategies to handle

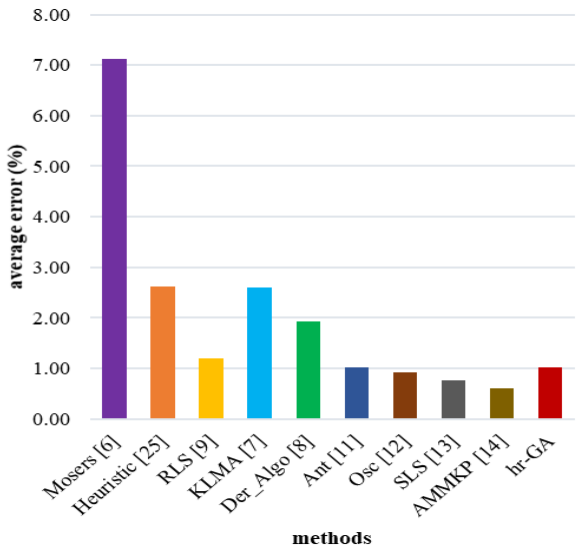

Figure 5. Comparison of the average errors between the heuristic methods

infeasible chromosomes, namely penalty and repairing procedure. Those are also hybridized a new simple local search technique to improve the solution quality. Several numerical experiments on a set of Benchmark instances taken from OR-Library have been conducted to evaluate the performance of the algorithms. The results show that repairing-based GA has better performance than penalty-based GA. Hybridizing GA with local search has also improved the solution quality immensely. The comparisons with other heuristic algorithms are also made based on the solution quality, the number of optimal solutions obtained, and average errors. It concludes that hr-GA is able to get competitive results with the other heuristics. These findings add to a growing body of literature on the applications of GA.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Mitsuo Gen gives the idea for the Algorithm; Admi Syarif and Dian Anggrainii designed the experiments; Dian Anggraini performed the experiments; Admi Syarif and Kurnia Muludi supervised the study, analyzed the results, verified the findings of the study, and wrote the paper.

Acknowledgments

The authors are grateful to "Hibah Unggulan" Lampung University, Grant-in-Aid No. 1514/UN26.21/PN/2020, 2020, for assisting this research. We thank the anonymous reviewers for their constructive comments that helped us a lot in substantially improving this paper.

References

[1] J. Gottlieb, "Permutation-based evolutionary algorithms for multi-dimensional knapsack problems", In: *Proc. of the 2000 ACM Symposium on Applied Computing - Volume 1 (SAC '00)*, New York, USA, Vol. 1, pp. 408–414, 2000.

[2] Y. Chen and J. K. Hao, "A 'reduce and solve' approach for the multiple-choice multi-dimensional knapsack problem", *European Journal of Operational Research*, Vol. 239, No. 2, pp. 313–322, 2014.

[3] H. Shojaei, A. H. Ghamarian, T. Basten, M. Geilen, S. Stuijk, and R. Hoes, "A parameterized compositional multi-dimensional multiple-choice knapsack heuristic for CMP run-time management", In: *Proc. of the 46th Annual Design Automation Conf. (DAC '09)*, New York, USA, pp. 917–922, 2009.

[4] H. Shojaei, T. H. Wu, A. Davoodi, and T. Basten, "A Pareto-algebraic framework for signal power optimization in global routing", In: *Proc. of the International Symposium on Low Power Electronics and Design*, New York, USA, pp. 407–412, 2010.

[5] L. Arockiam and N. Sasikaladevi, "Simulated Annealing Versus Genetic Based Service Selection Algorithms", *International Journal of u- and e- Service, Science and Technology*, Vol. 5, No. 1, pp. 35–50, 2012.

[6] M. Moser, D. P. Jokanovic, and N. Shiratori, "An algorithm for the multi-dimensional multiple-choice knapsack problem", *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, Vol. E80-A, No. 3, pp. 582–589, 1997.

[7] S. Khan, K. F. Li, E. G. Manning, and M. M. Akbar, "Solving the Knapsack Problem for Adaptive Multimedia Systems", *Studia Informatica Universalis*, Vol. 2, pp. 157–178, 2002.

[8] M. Hifi, M. Michrafy, and A. Sbihi, "Heuristic algorithms for the multiple-choice multi-dimensional knapsack problem", *Journal of the Operational Research Society*, Vol. 55, No. 12, pp. 1323–1332, 2004.

[9] M. Hifi, M. Michrafy, and A. Sbihi, "A reactive local search-based algorithm for the multiple-choice multi-dimensional knapsack problem", *Computational Optimization and Applications*, Vol. 33, No. 2–3, pp. 271–285, 2006.

[10] N. Cherfi and M. Hifi, "A column generation method for the multiple-choice multi-dimensional knapsack problem", *Computational Optimization and Applications*, Vol. 46, No. 1, pp. 51–73, 2010.

[11] S. Iqbal, M. F. Bari, and M. S. Rahman, "Solving the Multi-dimensional multi-choice knapsack problem with the help of ants", In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Berlin, Heidelberg, Vol. 6234, pp. 312–323, 2010.

[12] S. Htiouech, S. Bouamama, and R. Attia, "Using surrogate information to solve the multidimensional multi-choice knapsack problem", In: *Proc. of IEEE Congress on Evolutionary Computation*, Cancun, Mexico, pp. 2102–2107, 2013.

[13] Y. Xia, C. Gao, and J. L. Li, "A Stochastic Local Search Heuristic for the Multidimensional Multiple-choice Knapsack Problem", In: *Communications in Computer and Information Science*, Berlin, Heidelberg, Vol. 562, pp. 513–522, 2015.

[14] S. Htiouech and A. Alzaidi, "Smart Agents for the Multidimensional Multi-Choice Knapsack Problem", *International Journal of Computer Applications*, Vol. 174, No. 6, pp. 5–9, 2017.

[15] J. H. Holland, *Adaptation in Natural and Artificial Systems: an Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, University of Michigan Press, 1975.

[16] A. Syarif and M. Gen, "Solving exclusionary side constrained transportation problem by using a hybrid spanning tree-based genetic algorithm", *Journal of Intelligent Manufacturing*, Vol. 14, No. 3–4, pp. 389–399, 2003.

[17] Hiassat, A. Diabat, and I. Rahwan, "A genetic algorithm approach for location-inventory-routing problem with perishable products", *Journal of Manufacturing Systems*, Vol. 42, pp. 93–103, 2017.

[18] M. Gen and L. Lin, "Multiobjective evolutionary algorithm for manufacturing scheduling problems: State-of-the-art survey", *Journal of Intelligent Manufacturing*, Vol. 25, No. 5, pp. 849–866, 2014.

[19] M. A. Mohammed, M. K. Abd Ghani, R. I. Hamed, S. A. Mostafa, M. S. Ahmad, and D. A. Ibrahim, "Solving vehicle routing problem by using improved genetic algorithm for the optimal solution", *Journal of Computational Science*, Vol. 21, pp. 255–262, 2017.

[20] A. H. C. Van Kampen, C. S. Strom, and L. M. C. Buydens, "Lethalization, penalty and repair functions for constraint handling in the genetic algorithm methodology", *Chemometrics and Intelligent Laboratory Systems*, Vol. 34, No. 1, pp. 55–68, 1996.

[21] M. Gen and A. Syarif, "Hybrid genetic algorithm for multi-time period production/distribution planning", *Computers and Industrial Engineering*, Vol. 48, No. 4, pp. 799–809, 2005.

[22] S. Hou, Y. Liu, H. Wen, and Y. Chen, "A self-crossover genetic algorithm for job shop scheduling problem", In: *Proc. of IEEE International Conf. on Industrial Engineering and Engineering Management*, pp. 549–554, 2011.

[23] M. Gen and R. Cheng, *Genetic Algorithms and Engineering Optimization*, John Wiley & Sons, New York, 2000.

[24] W. Wan and J. B. Birch, "An improved hybrid genetic algorithm with a new local search procedure", *Journal of Applied Mathematics*, Vol. 2013, 2013.

[25] ftp://cermsem.univ-paris1.fr/pub/CERMSEM /hifi/MMKP/ (accessed Jul. 15, 2019).