Bulletin of Networking, Computing, Systems, and Software – www.bncss.org, ISSN 2186-5140
Volume 8, Number 2, pages 129–132, July 2019

A strawberry optimization algorithm for the multi-objective knapsack problem
(preliminary version)

Yamato Mizobe Akihiro Fujiwara
Graduate School of Computer Science and Systems Engineering
Kyushu Institute of Technology
Iizuka, Fukuoka, 820-8502, Japan

***Abstract*—A multi-objective optimization problem involves a nubmer of objective functions to be maximized or minimized, and no single solution exists for the problem because there is no solution that simultaneously satisfies all of the objective functions. Therefore, a set of Pareto-optimal solutions, which are non-dominated solutions for the problem, is defined for the multi-objective optimization problem.**

In the present paper, we consider multi-objective knapsack problem, which is one of well-known multi-objective optimization problems, and propose an optimization algorithm based on a strawberry algorithm (SBA). The experimental results show that the proposed algorithm obtains a better set of Pareto solutions than the existing algorithm.

***Index Terms*—multi-objective optimization, knapsack problem, strawberry algorithm**

I. Introduction

A multi-objective optimization problem involves a number of objective functions to be maximized or minimized. The multi-objective optimization problem has been considered in a number of fields, such as economics or engineering, and the most optimal solution is needed for the practical problems.

However, there is no single solution for the multi-objective optimization problem because no solution satisfies all of objectives simultaneously. Since there exists a trade-off between two or more conflicting objective functions, the non-dominated solutions, which are not inferior to the other solutions in all of the objective functions, are needed for the multi-objective optimization problem.

The non-dominated solution is called Pareto optimal solution [1], and a set of maximal Pareto optimal solutions is considered as an optimal solution for the multi-objective optimization problem. Since the problem for computing the maximal Pareto optimal solutions for the multi-objective optimization problem is generally computationally hard, a number of approximation algorithms [2], [3], [6] have been proposed for the problem.

As an example of the multi-objective optimization problems, a number of approximation algorithms have proposed for a multi-objective 0-1 knapsack problem. A standard 0-1 knapsack problem [4] is generally given with a knapsack and multiple items, and value and weight are defined for each item. On the other hand, there exists a number of knapsacks in the multi-objective 0-1 knapsack problem, and values and weights of items are defined for each knapsack.

A number of approximation algorithms have been proposed for the multi-objective knapsack problem. Deb et. al. [5] proposed an approximation algorithm based on genetic approach. In addition, some algorithms have been proposed based on a group intelligence optimizations, such as bee colony optimization [6] and particle swarm optimization and a firefly algorithm [7].

In the present paper, we propose an approximation algorithm for the multi-objective 0-1 knapsack problem using strawberry algorithm (SBA) [9]. The strawberry optimization is an optimization method based on the ecology of strawberry. The strawberry develops runners and roots for searching water resources and minerals, and the runners and roots are used for global and local searches in the optimization method.

We implement our proposed algorithm and an existing algorithm [5] in experimental environment, and evaluate validity of the proposed algorithm. the experimental results show that our proposed algorithm obtains a better set of Pareto solutions than the existing algorithm.

II. Preliminaries

A. Multi-objective optimization problem

In this section, we first define the multi-objective optimization problem. We assume that an instance of the problem is k-dimensional decision vector $\boldsymbol{x}$. The multi-objective optimization problem consists of a set of n objective functions $\{f_0(\boldsymbol{x}), f_1(\boldsymbol{x}), \cdots, f_{n-1}(\boldsymbol{x})\}$ and a set of k constraint functions $\{g_0(\boldsymbol{x}), g_1(\boldsymbol{x}), \cdots, g_{k-1}(\boldsymbol{x})\}$. Then, each objective function is defined as image from $\boldsymbol{x}$ to n objective function vector $\boldsymbol{y}$. The definition is mathematically formulated as follows.

$$\begin{aligned} \max / \min \ \ & \boldsymbol{y} = \{y_0, y_1, ..., y_{n-1}\} = \{f_0(\boldsymbol{x}), f_1(\boldsymbol{x}), ..., f_{n-1}(\boldsymbol{x})\} \\ \text{such that } & \boldsymbol{x} = (x_0, x_1, ..., x_{m-1}) \in X, \\ & X = \{\boldsymbol{x} \mid \forall i \in \{0, 1, ..., k-1\}, g_i(\boldsymbol{x}) \leq 0\} \end{aligned}$$

In the above definition, X is called as a set of feasible solutions for the problem.

Since no solution satisfies all of the objective functions in the multi-objective optimization problem, a solution that is not

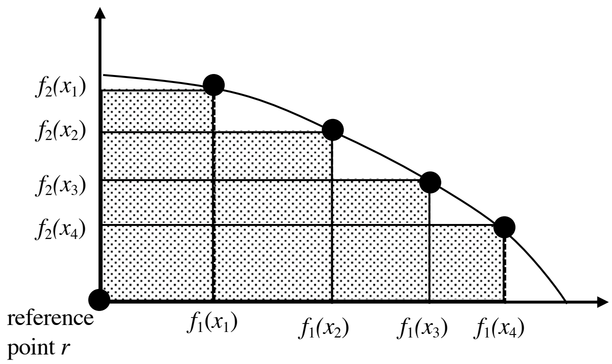

Fig. 1. An example of Hypervolume indicator

inferior to the other solutions is needed for the multi-objective problem. The solution is called *Pareto optimal solution*, and we first define the dominance relationship of the solutions for defining the Pareto-optimal solution.

We assume that $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ are two feasible solutions for the problem and all objective functions are maximized. Then, $\boldsymbol{x_2}$ dominates $\boldsymbol{x_1}$ if and only if the following two conditions holds.

$$\forall i \in \{0, 1, ..., n-1\}, f_i(\boldsymbol{x}_1) \leq f_i(\boldsymbol{x}_2)$$
$$\exists j \in \{0, 1, ..., n-1\}, f_j(\boldsymbol{x}_1) < f_j(\boldsymbol{x}_2)$$

In this paper, $\boldsymbol{x}_1 \prec \boldsymbol{x}_2$ denotes $\boldsymbol{x}_2$ dominates $\boldsymbol{x}_1$,

In addition, a feasible solution $\boldsymbol{x}$ is called *Pareto optimal* if and only if there is no feasible solution $\boldsymbol{x'} \in X$ such that $\boldsymbol{x} \prec \boldsymbol{x'}$. Since the Pareto-optimal solution is a solution that cannot be improved in any of the objective function without degrading one of the other functions, a maximal set of the Pareto optimal solution is considered as an optimal solution for the multi-objective optimization problem.

There are various metrics for a set of Pareto-optimal solutions for the multi-objective optimization problem. In the paper, we evaluate a set of the Pareto-optimal solutions using *hypervolume (HV) indicator* [8].

Let v_x be the volume of the hypercube created by Pareto optimal solution $\boldsymbol{x}$ and the reference point $\boldsymbol{r}$. The hypervolume V for a set of Pareto optimal solutions is defined as follows.

$$V = \bigcup_{\boldsymbol{x} \in X} v_x$$

Fig. 1 shows an example of the hypervolume indicator in case that the problem is bi-objective. Let $X = \{\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3}, \boldsymbol{x_4}\}$ be a set of Pareto optimal solutions. The hypervolume for X is defined as an union, which is an gray-shaded area in the figure, of rectangular regions whose opposite vertices are $(f_1(\boldsymbol{x_i}), f_2(\boldsymbol{x_i}))$ and a reference point $\boldsymbol{r}$.

III. Strawberry optimization algorithm for the multi-objective knapsack problem

In this section, we first show definition of the multi-objective 0-1 knapsack problem, which is a well-known multi-objective optimization problem. We next explain an outline of the strawberry algorithm, and finally show our strawberry optimization algorithm for the knapsack problem.

A. The multi-objective 0-1 knapsack problem

An input of the multi-objective 0-1 knapsack problem is given as follows.

- n knapsacks whose capacities are $c_0, c_1, \cdots c_{n-1}$.
- m items stored in the knapsacks. $v_{i,j}$ and $w_{i,j}$ denote value and weight of item j for knapsack i, respectively.

Let $_x = (x_0, x_1, \cdots, x_{m-1})$ are m-dimensional Boolean vector. Then, the multi-objective 0-1 knapsack problem is formulated as follows.

$$\max \ \boldsymbol{y} = \{f_0(\boldsymbol{x}), f_1(\boldsymbol{x}), ..., f_{n-1}(\boldsymbol{x})\}$$
$$f_i(\boldsymbol{x}) = \sum_{j=0}^{m-1} p_{i,j} x_j$$
$$\text{such that } g_i(\boldsymbol{x}) = \left(\sum_{j=0}^{m-1} w_{i,j} x_j \right) - c_i \leq 0$$

In the above expression, $x_j = 1$ denotes item j is stored in the knapsacks.

B. An outline of strawberry algorithm

Strawberry algorithm (SBA) is an optimization method based on ecology of strawberry. Strawberry develop runners and roots for searching water resources and minerals. A function of the runner is to search for resources away from a parent turnip, and the function allows the strawberry to breed efficiently. On the other hand, the root of the strawberry is used to search and absorb nearby nutrition. Using the feature of the strawberry, global and local searches can be simultaneously performed for tentative solutions.

We now outline the strawberry algorithm. The algorithm consists of an initial part and a repeated part. The repeated part is executed repeatedly by the number of a given parameter.

Initial part:

We assume there are l parent plants at the beginning of the algorithm. Each parent plant stores an initial solution obtained by a trivial algorithm. We assume a set of parent plants is $T = \{t_0, t_1, \cdots, t_{l-1}\}$ and each t_p stores an initial solution $x_p(0)$.

Repeated part:

Two m-dimensional Boolean vectors d_{runner} and d_{root}, which denote distances from the parent plant, are generated. We assume that $d_{runner} = (ru_0, ru_1, \cdots ru_{m-1})$ and $d_{root} = (ro_0, ro_1, \cdots ro_{m-1})$, and two vectors are used for global and local search, respectively. Distances for the two vectors are denoted by the numbers of value "1" in the two Boolean vectors.

Let $x_p(q) = (x_0, x_1, \cdots, x_{m-1})$ a solution stored in t_p. Then, two tentative solutions, y_{2p} and y_{2p+1}, are created using d_{runner} and d_{root} for each $x_p(q)$ as follows.

$$y_{2p} = (x_0 \oplus ro_0, x_1 \oplus ro_1, \cdots, x_{m-1} \oplus ro_{m-1})$$
$$y_{2p+1} = (x_0 \oplus ru_0, x_1 \oplus ru_1, \cdots, x_{m-1} \oplus ru_{m-1})$$

In other words, each x_k is inverted if ro_k or ru_k is 1, and two results of EX-OR between $x_p(q)$ and two vectors, d_{runner} and d_{root} are modified solutions with global and local search, respectively.

Since two tentative solutions are created for each parent plant, $2l$ tentative solutions $y_0, y_1, \cdots, y_{2l-1}$ are obtained after the above step.

Finally, $2l$ tentative solutions are sorted according to evaluation values of the solutions, and l best solutions are selected as $x_0(q+1), x_1(q+1), \cdots, x_{l-1}(q+1)$, which denote l parent plants in the next step.

C. SBA for multi-objective knapsack problem

We now propose an optimization algorithm for the multi-objective knapsack problem using SBA. In the optimization algorithm using SBA, infeasible solution may be created in a repeated part of SBA. In case of the infeasible folution, a number of items are removed from a tentative solution using greedy method.

We show our proposed algorithm for the multi-objective knapsack problem using SBA. The algorithm consists of three steps.

Algorithm: an algorithm for multi-objective 0-1 knapsack problem using SBA

Step 1: For each parent plant t_p $(0 \leq p \leq l-1)$, generate m-dimensional Boolean vector $x_p(0) = (x_0, x_1, \cdots, x_{m-1})$ as an initial solution. Each x_k is set to 0 or 1 with probability $\frac{1}{2}$. In addition, set $P = \phi$. (P is a set of Pareto optimal solution.)

Step 2: The following sub-steps are repeated G times. (G is a parameter that denotes the number of generations.)

(2-1) Two m-dimensional Boolean vectors d_{root} and d_{runner}, which denote distances from tentative solution are generated. In these two vectors, the number and positions of value "1" are decided by an uniform distribution. The range of numbers of "1" is $[1, 3]$ for d_{root} and $[\frac{m}{2}, m]$ for d_{runner}.

(2-2) For each parent plant t_p $(0 \leq p \leq l-1)$, two tentative solutions, y_{2p} and y_{2p+1}, are created using d_{runner}, d_{root} and $x_p(q)$.

$$\begin{aligned} y_{2p-1} &= (x_0 \oplus ro_0, x_1 \oplus ro_1, \cdots, x_{m-1} \oplus ro_{m-1}) \\ y_{2p} &= (x_0 \oplus ru_0, x_1 \oplus ru_1, \cdots, x_{m-1} \oplus ru_{m-1}) \end{aligned}$$

(2-3) In case that y_{2p} or y_{2p+1} is infeasible solution, that is, one of constraints for the knapsack is not satisfied, the following sub-steps are repeated until all constraints are satisfied.

(2-3-1) For each j $(0 \leq j \leq m-1)$, compute s_j as follows.

$$s_j = \sum_{i=0}^{n-1} \frac{p_{k,j}}{w_{k,j}}$$

(2-3-2) Select x_j that satisfies $x_j = 1$ and the following condition.

$$s_j = \min\{s_k \mid x_k = 1, 0 \leq k \leq m-1\}$$

(2-3-2) Set $x_j = 0$.

(2-4) Set $P' = P \cup \{y_0, y_1, \cdots, y_{2l-1}\}$ and $r = 1$. Then, compute Pareto rank of each solution in P' by repeating the following sub-steps until $P' = \phi$.

(2-4-1) Select Pareto optimal solutions in P', and set Pareto ranks of the selected solutions to r.

(2-4-2) Remove the selected solutions from P', and set $r = r + 1$.

(2-5) Sort all solutions in $P \cup \{y_0, y_1, \cdots, y_{2l-1}\}$ according to the computed Pareto ranks. Then, select l best solutions as parent plant $t_0, t_1, \cdots, t_{l-1}$. In addition, all solutions whose rank is 1 is selected as a set of Pareto optimal solutions P.

Step 3: Output P as a set of Pareto optimal solutions.

IV. Experimental results

Our proposed algorithm and an existing algorithm [5] are implemented using C++, and we compare Pareto optimal solutions and hypervolume indicators.

First, we describe details of the multi-objective 0-1 knapsack problem. The values of the variables used in the simulation are as follows.

- The number of knapsacks n: 2
- The number of items m: 500
- A value of item $p_{i,j}$ $(1 \leq i \leq n, 1 \leq j \leq m)$: a randomly generated integer in [10, 100]
- A weight of item $w_{i,j}$ $(1 \leq i \leq n, 1 \leq j \leq m)$: a randomly generated integer in [10,100]
- A capacity of a knapsack c_i $(1 \leq i \leq n)$: $c_i = \frac{1}{2}\sum_{j=1}^{m} w_{i,j}$

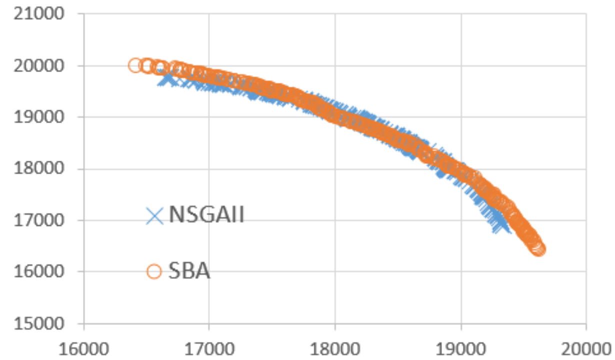

Fig. 2. Pareto optimal solutions of the algorithms

Fig. 2 shows a part of our experimental results. Pareto optimal solutions obtained by the proposed algorithm is distributed in wider range than the solutions obtained by the existing algorithm.

Table I shows hypervolumes of the algorithms. Since the value of the proposed algorithm is better than the existing algorithm, our proposed algorithm obtains a better set of Pareto solutions than the existing algorithm.

TABLE I
HYPERVOLUMES OF THE ALGORITHMS

The proposed algorithm	The existing algorithm [5]
3.89×10^8	3.80×10^8

V. Conclusions

In this paper, we proposed an approximation algorithm for the multi-objective 0-1 knapsack problem using SBA. We compared our proposed algorithm and an existing algorithm in experimental environment, and showed validity of the proposed algorithm from the viewpoint of hypervolume indicator.

As our future research, we are considering improvement of our proposed algorithm for distribution of the Pareto optimal solutions. We also considering reduction of the execution time of the proposed algorithm.

VI. Acknowledgments

This research was partially supported by JSPS KAKENHI, Grand-in-Aid for Scientific Research (C), 16K00021.

References

[1] E. Zitzler and L. Thiele, "Multiobjective optimization using evolutionary algorithmsa comparative case study, " *International conference on parallel problem solving from nature*, 1998, pp. 292–301.

[2] C. Bazgan, H. Hugot and D. Vanderpooten, "Solving efficiently the 0-1 multi-objective knapsack problem," *Computers and Operations Research*, Vol. 36, No. 1, 2009, pp. 260–279.

[3] E. Zitzler and L. Thiele, "Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach, " *IEEE Transactions on Evolutionary Computation*, Vol. 3, No. 4, November, 1999, pp. 257–271.

[4] H. Kellerer, U. Pferschy and D. Pisinger, "Knapsack Problems, " *Springer*, 2004.

[5] K. Deb, S. Agrawal, A. Pratap and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II, " *International conference on parallel problem solving from nature*, 2000, pp. 849–858.

[6] S. Sabet, M. Shokouhifar and F. Farokhi, " A discrete artificial bee colony for multiple Knapsack problem, " *International Journal of Reasoning-based Intelligent Systems*, Vol. 5, No. 2, 2013, pp. 88–95.

[7] D. Zouache, A. Moussaoui and F. B. Abdelaziz, " A cooperative swarm intelligence algorithm for multi-objective discrete optimization with application to the knapsack problem, " *European Journal of Operational Research*, Vol. 264, No. 1, 2018, pp. 74–88.

[8] A. Auger, J. Bader, D. Brockhoff and E. Zitzler, " Hypervolume-based multiobjective optimizatio: Theoretical foundations and practical implications, " *Theoretical Computer Science*, Vol. 425, No. 30, 2012, pp. 75–103.

[9] F. Merrikh-Bayat, "A Numerical Optimization Algorithm Inspired by the Strawberry Plant, " *arXiv preprint arXiv:1407.7399*, 2014.