Adaptive landscape-aware constraint handling with application to binary knapsack problems

Arnaud Liefooghe, Katherine M. Malan

Adaptive Landscape-aware Constraint Handling with Application to Binary Knapsack Problems

Arnaud Liefooghe
arnaud.liefooghe@univ-lille.fr
Univ. Lille, CNRS, Inria, Centrale Lille, UMR 9189 CRIStAL
F-59000 Lille, France

Katherine M. Malan
malankm@unisa.ac.za
University of South Africa
Pretoria, South Africa

ABSTRACT

Real-world optimization problems frequently have constraints that define feasible and infeasible combinations of decision variables. Evolutionary algorithms do not inherently work with constraints, so they must be modified to include a suitable constraint handling technique (CHT) before they can be used to solve such problems. A range of different approaches to handling constraints have been used effectively with evolutionary algorithms, such as penalty-based, repair, feasibility rules, and bi-objective approaches. In this study we investigate different CHTs with an evolutionary algorithm on the 0/1 knapsack problem. We present results to show that performance complementarity exists between different CHTs on the knapsack problem. Landscape analysis is then used to characterize the search space of a large number of knapsack instances, and decision tree induction is used to derive rules for selecting the most appropriate CHT and switching it adaptively based on landscape features. We finally implement a landscape-aware CHT approach and show that it out-performs the constituent CHT approaches.

CCS CONCEPTS

• **Theory of computation** → *Random search heuristics*; • **Applied computing** → *Multi-criterion optimization and decision-making*.

KEYWORDS

Landscape-aware search, landscape analysis, constrained search spaces, knapsack problem.

ACM Reference Format:
Arnaud Liefooghe and Katherine M. Malan. 2023. Adaptive Landscape-aware Constraint Handling with Application to Binary Knapsack Problems. In *Genetic and Evolutionary Computation Conference Companion (GECCO '23 Companion), July 15–19, 2023, Lisbon, Portugal.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3583133.3596405

This research and collaboration was inspired from attending the Lorentz Center workshop *Benchmarked: Optimization Meets Machine Learning* 30 May - 3 June 2022 in Leiden, the Netherlands.
K.M.M. supported by the National Research Foundation of South Africa CPRR Grant Number 120837.

1 INTRODUCTION

For many years of GECCO conferences, there was a focus on benchmarking the performance of evolutionary algorithms on numerical problems without constraints (except for boundary constraints). The state-of-the-art evolutionary algorithms were deemed to be the ones that performed the best on the BBOB (Black Box Optimization Benchmarking) test suite of that year. Recognizing the importance of constraints, this focus has started to shift. For GECCO 2022, the organizers of the BBOB workshop invited "in particular contributions related to constrained optimization" [2].

In the real world, optimization problems with constraints are more common than those without. In the GECCO 2022 track on *Real World Applications*, there were 13 papers accepted that had optimization as the main component, and more than half of the models involved complex constraints over the search space. For example, one study addressing the optimal design of execution plans of robots [8] described a model with two objectives and 11 constraints. With these kinds of problems, we argue that the nature of the violation landscape [16] formed by the constraints can have as much effect (if not more effect) on the difficulty for search as the fitness landscape formed by the objective(s).

There are a number of established approaches to handling constraints with evolutionary algorithms [4, 21], including repair of solutions, penalty approaches, feasibility rule approaches, and treating the constraint(s) as additional objective(s), with feasibility rules being the most popular approach with nature-inspired algorithms [20]. It has been shown that using an ensemble of constraint handling techniques in parallel with an evolutionary algorithm out-performs the individual constraint handling approaches [17]. Our proposed approach builds on this by also including landscape information to guide the choice of the CHT.

We argue that the choice of the most appropriate CHT depends on the particular problem instance and the way in which the constraints modify the feasible space. For example, an approach that rejects infeasible solutions (known as death penalty) will perform poorly when the problem has a low proportion of feasible solutions [21]. In these cases, an approach using feasibility rules is more effective [15]. Penalty approaches in general are also not suitable for problems where the best solution is located on the boundary of the feasible region or when the feasible regions are disjoint [18], but are suitable when the fitness and level of violation are positively correlated [15].

Previous work has shown the effectiveness of a landscape-aware approach to constraint handling for evolutionary algorithms in continuous search spaces [13]. In this paper we take a similar approach applied to binary search spaces by studying the behavior of different CHTs with an evolutionary algorithm on a wide range of

knapsack problems with varying difficulty. More particularly, our contributions are three-fold:

(i) We first confirm that there is indeed performance complementarity between different CHTs when used with an evolutionary algorithm. We achieve this through large-scale benchmarking, covering numerous knapsack problem instances with different characteristics.
(ii) We then compute a number of features characterizing the violation and the fitness landscape induced by the constrained search space, and we show that some features correlate with the performance of CHTs.
(iii) We finally inject the landscape features into the evolutionary search process in order to adjust the CHT approach at runtime, and we show that this adaptive landscape-aware approach leads to an improvement in performance.

The rest of the paper is organized as follows. In Section 2, we introduce the knapsack problem. In Section 3 we present a number of CHTs and we report their experimental performance over a number of instances. In Section 4, we perform a landscape analysis for the same set of knapsack problem instances. In Section 5, we introduce the proposed landscape-aware approach and we discuss its performance. In the last section, we conclude the paper and discuss further research.

2 THE 0/1 KNAPSACK PROBLEM

The knapsack problem is a well-known and extensively studied NP-hard problem with application in real-world contexts such the packing and loading of goods [19]. The basic 0/1 knapsack problem with n items can be formulated as follows:

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{n} p_i x_i, \\ \text{subject to} \quad & \sum_{i=1}^{n} w_i x_i \ \leq \ W, \end{aligned} \tag{1}$$

where x_i is a binary variable indicating whether or not item $i \in \{1, \ldots, n\}$ is included in the knapsack, p_i is the profit gained by item i, w_i is the weight of item i, and W is the capacity weight of the knapsack.

Early work on solving the 0/1 knapsack problem using a genetic algorithm with different CHTs was performed by Michalewicz and Arabas [22]. They investigated three types of penalty-based approaches: logarithmic, linear and quadratic penalty with respect to the degree of violation. On problem instances that had an average knapsack capacity, the logarithmic penalty approach performed better than the other two penalty-based approaches, but none of the penalty-based approaches performed well on instances with a restrictive knapsack capacity. Overall, they found that a repair approach was more effective at finding optimal solutions than a penalty-based approach — a result that was confirmed in our preliminary experiments.

3 PERFORMANCE COMPLEMENTARITY

The success of any form of automated algorithm selection depends on the existence of *performance complementarity* [12] between the constituent algorithms. Performance complementarity refers to the situation where each algorithm has a performance advantage on different problem instances. If there is no performance complementarity between algorithms, it means that one algorithm is the best on all problem instances under study, and this implies that there would be no benefit to implementing automated algorithm selection. In a study of CHTs with evolutionary algorithms [15], it was shown that there was evidence of performance complementarity of CHTs on a combined set of 142 continuous and combinatorial problem instances. We start this study by showing that this is also the case on knapsack problem instances.

3.1 Problem Instances

We generate 1 080 knapsack problem instances with the following parameters:

n: The number of items that are available, n is set to 1 000. The search space size (number of binary strings) is thus $2^{1\,000} \approx 10^{300}$ solutions.
ρ: The correlation between the profits and weights, ρ, influences the difficulty of the problem [22]. Instances are generated with nine correlation values from −0.9 to +0.9: $\rho \in \{-0.9, -0.7, -0.4, -0.2, 0.0, 0.2, 0.4, 0.7, 0.9\}$.
W: The capacity of the knapsack, W, is defined as a proportion, c, of the sum of weights of all items:

$$W = c \cdot \sum_{i=1}^{n} w_i \tag{2}$$

Lower c-values result in a more constrained problem, so the optimal solution will consist of fewer items than instances with higher values of c. For example, $c = 0.6$ implies that adding more than 60% of the items on average would be infeasible, while $c \geq 1$ represents the trivial case of all solutions being feasible. For the instance generation, the value of c ranges from 0.05 to 0.6 in increments of 0.05.
s: The random seed, s. For each combination of ρ and c, 10 random seeds are used to independently generate 10 problem instances with random weights and profits.

In total, 1 080 training problem instances are generated (9 values of $\rho \times 12$ values of $c \times 10$ random seeds).

3.2 Constraint Handling Techniques

This section describes the base evolutionary algorithm used in the experiments as well as the four selected CHTs: linear penalty, quadratic penalty, feasibility rules, and a bi-objective approach.

Although initial experimentation showed that a repair strategy was very often more effective than the other considered CHTs on the knapsack problem, repair was not included in the portfolio of CHTs. The reason for excluding repair is because it relies on problem-specific knowledge and is therefore not a general strategy that could be applied to other problems, in particular black-box problems. In addition, unlike the trivial repair operation of knapsack (removing items from the knapsack until the weight is not exceeded), repair strategies of some problems can be more complex than solving the problem itself [22], so are not always feasible to implement. Initial experimentation also showed that a death penalty [21] and a logarithmic penalty [22] approach performed

badly over all instances, so these two penalty approaches were also discarded from the portfolio of CHTs.

3.2.1 Base Algorithm. We consider a simple steady-state (10 + 1) evolutionary algorithm that starts with a random population of size 10. At each iteration, two parents are independently selected from the population by means of binary tournament. Next, an offspring is generated with one-point crossover followed by stochastic mutation such that each binary variable is flipped with a rate of $\frac{1}{n}$. In case the offspring does not already belong to the population, it replaces the worst solution from the population if there is an improvement with respect to the considered CHT. This process is iterated until the maximum budget (in number of evaluations) is reached. The algorithm records and returns the best found solution in terms of constraint violation first, and profit value next. That is, a solution closer to the feasible region is always better, and among feasible solutions the one with the highest profit is preferred.

3.2.2 CHT1: Linear penalty (LP). In the case of a linear penalty approach, the extent to which the weight of the knapsack is exceeded is simply subtracted from the sum of the profits of the included items:

$$f(\mathbf{x}) = \begin{cases} \sum_{i=1}^{n} p_i x_i & \text{if } \mathbf{x} \text{ is feasible} \\ \sum_{i=1}^{n} p_i x_i - \left(\sum_{i=1}^{n} w_i x_i - W \right) & \text{otherwise} \end{cases} \tag{3}$$

3.2.3 CHT2: Quadratic penalty (QP). With the quadratic penalty approach, the penalty term is squared:

$$f(\mathbf{x}) = \sum_{i=1}^{n} p_i x_i - \left(\sum_{i=1}^{n} w_i x_i - W \right)^2 \quad \text{if } \mathbf{x} \text{ is infeasible} \tag{4}$$

3.2.4 CHT3: Feasibility rules (FR). Deb [5] proposed an alternative to a penalty-based approach for evolutionary algorithms that requires no parameter. The approach, termed *feasibility rules*, distinguishes between feasible and infeasible solutions using the following rules for pairwise comparisons:

(1) A feasible solution is preferred over an infeasible solution.
(2) If both solutions are feasible, the one with the higher fitness is preferred.
(3) If both solutions are infeasible, the one with the lowest extent of constraint violation is preferred.

3.2.5 CHT4: Bi-objective (BO). For the bi-objective approach, we simply consider violation and fitness (profit) as two separate objectives to be optimized simultaneously. To do so, we rely on the principles of the non-dominated sorting genetic algorithm II (NSGA-II) [6]. Selection is based on non-dominated sorting, such that dominating solutions are better. Among mutually non-dominated solutions, solutions that are in less crowded areas of the objective space are preferred. As such, the bi-objective CHT is expected to maintain a population with good and well-diversified trade-offs among profit and violation. Here as well, the final solution is the one following the lexicographic order of violation first, and profit next.

Table 1: Illustrative example of performance ranking of four CHTs on a single instance of the knapsack problem.

	Success rate	Mean profit	Mean violation	Rank
LP	0.9	75263.7	0.6	3
QP	1.0	75050.5	0	2
FR	1.0	75191.6	0	1
BO	0.0	n/a	25676.1	4

3.3 Setup and Performance Ranking

We performed 10 independent runs of each CHT approach on each of the 1 080 problem instances with a total budget of 100 000 evaluations per run. For each CHT the following performance metrics were recorded at function evaluation budget intervals of $\lceil 10^i \rceil$ for $i \in \{1.00, 1.25, 1.50, \ldots, 5.00\}$:

- Success rate: the proportion of runs that returned a final solution that is feasible within the given budget.
- Violation: the mean level of violation of all runs.
- Fitness: the mean profit of the best solutions of all runs that returned feasible solutions.

For each problem instance, at each budget interval, the CHTs were ranked using the CEC 2010 competition rules [24]:

(1) The algorithm with a higher success rate wins.
(2) Given an equal success rate greater than zero, the algorithm with a higher fitness value wins.
(3) Given an equal success rate = 0, the algorithm with the lower violation wins.

To illustrate the ranking, Table 1 shows an example of the performance metrics on a sample instance. QP and FR both found feasible solutions in all 10 runs, resulting in a success rate of 1. Comparing based on profit, FR wins and is given the rank of 1. Although the mean profit value of LP is higher than those of QP and FR, it is given the rank 3, because it did not return feasible solutions in all runs. BO did not return feasible solutions in any of the runs, so is given the rank of 4.

3.4 Experimental Results

Figure 1 shows the average relative performance of the four CHTs over all training instances. From left to right the graphs show: (1) the average rank – lower is better, (2) the proportion of times that the considered CHT was one of the best performing strategies, (3) the proportion of times that the CHT was one of the worst performing strategies.

From the results in Figure 1, we can see there is evidence of performance complementarity between the four CHTs. The bi-objective approach is the best performing strategy with a budget below 1 000 evaluations (evident in the low rank and high proportion of instances on which it is the best strategy), but then moves to the worst performing strategy on the larger budget of 100 000 evaluations. The other three strategies performed similarly to each other, with the linear penalty performing the worst on a larger number of instances than the other two approaches. Quadratic penalty and feasibility rules perform quite similarly at the early stages of

the runs, while the former eventually out-performs the latter on average for larger budgets.

Table 2 shows the mean rank as well as the number of instances on which each strategy was both the worst and the best performing approach after a small and large budget of evaluations. From Table 2 we can see that QP achieved the lowest (i.e. best) mean rank at both the low and high budget. However, we also see that QP was the worst performing approach on 145 of the instances at a budget of 1 000, providing further evidence of performance complementarity.

4 LANDSCAPE ANALYSIS

There are many established techniques for characterizing different features of continuous and combinatorial optimization problems [14]. However, these techniques typically assume that the problem is unconstrained. When problems have constraints, there are additional features that can be used to characterize the violation landscape as well as the interaction between the fitness and violation landscapes [15]. This section defines the five landscape metrics used in this study as well as the approach used to sample and characterize the knapsack instances.

4.1 Constrained Landscape Features

The features described in this section were originally proposed by Malan et al. [16].

4.1.1 Feature 1: Feasibility ratio (`fsr`). Given a sample of solutions S, `fsr` is defined as the proportion of solutions in S that are feasible.

4.1.2 Feature 2: Ratio of feasible boundary crossings (`rfbx`). The `rfbx` feature provides a measure of the degree of disjointedness of the feasible areas in the search space. Given a walk through the search space consisting of a sequence of solutions, a step in the walk is regarded as crossing a boundary if the feasibility status changes from one solution to the next. `rfbx` is defined as the proportion of steps in the walk that cross feasible boundaries.

4.1.3 Feature 3: Fitness violation correlation (`fvc`). A sample S of solutions to the knapsack problem defines $|S|$ profit-vs-violation pairs, where the violation is the extent to which the knapsack weight is exceeded. The subset S_{inf} consists of the solutions in S that are infeasible. The `fvc` feature is defined as the Spearman's correlation coefficient between the profit values (to be maximized) and violation values (to be minimized) of solutions in S_{inf}. If S_{inf} is an empty set, `fvc` is set to 0.

4.1.4 Feature 4: Proportion of solutions in ideal zone 25 (`piz25`). Given a sample of solutions S, with associated profit and violation values. The 25% *ideal zone* is defined as the solutions that have a profit value above the median of the profit values in the sample as well as a violation value below the median violation value of the sample. The `piz25` feature is defined as the proportion of solutions in sample S that fall into this 25% ideal zone.

4.1.5 Feature 5: Proportion of solutions in ideal zone 4 (`piz04`). Given a sample of solutions S, with associated profit and violation values. The 4% *ideal zone* is defined as the solutions that have a profit value above the 20^{th} percentile of the profit values in the sample as well as a violation value below the 20^{th} percentile of the violation values of the sample. The `piz04` feature is defined as the proportion of solutions in sample S that fall into this 4% ideal zone.

4.2 Sampling Strategy

The sequence of solutions S necessary to compute the above features is simply collected by means of the considered evolutionary algorithm. Given an algorithm run under a CHT, we add a solution to S each time there is an improvement, that is, when the corresponding offspring is accepted into the population. As such, the sample size does not exceed the number of solutions visited by the algorithm. We end up with a different sequence for each run of each CHT, and at different time steps. We first compute the feature values for each scenario, and thus we compute the average feature values over the runs. We expect such a sampling strategy to better capture the dynamics of the evolutionary algorithm. Moreover, the walk does not induce any additional cost to the training phase on top of actually running the evolutionary algorithm with each CHT.

4.3 Characterization of Training Instances

Based on evolutionary algorithm samples, all training instances were characterized using the five landscape features described in Section 4.1. Figure 2 shows the distribution of values for each metric over all of the instances for different budgets.

The values for `fsr` start with values close to 0 at a low budget. With an increase in budget, the sampling based on feasibility rules and quadratic penalty finds a larger portion of feasible solutions, while the other two strategies sample more solutions in infeasible regions. The values for `rfbx` are higher for the linear and bi-objective strategies corresponding with the search in infeasible regions. Overall, the low values for `rfbx` for the knapsack problem indicate that the regions of feasibility are contiguous.

The values of `fvc` filled the full range from -1 to 1, with most of the strategies finding samples of solutions with a positive correlation between fitness and violation, except for linear penalty that found samples with negative correlation. For the ideal zone metrics, the medians of `piz25` and `piz04` for all strategies except bi-objective were all quite high (above 0.4 for `piz25` and in the region of 0.1 and 0.2 for `piz04`). This indicates that the samples included large proportions of solutions in the ideal zones, as an even distribution of solutions in the fitness-violation space would result in a value of 0.25 for `piz25` and a value of 0.04 for `piz04`.

5 LANDSCAPE-AWARE CONSTRAINT HANDLING

In this section, we present the adaptive landscape-aware constraint handling approach proposed based on our previous observations.

5.1 Offline Training

A training set was created for each CHT strategy (LP, QP, FR, and BO) consisting of the 1 080 problem instances with their associated five landscape metrics calculated based on the sample generated by the evolutionary algorithm with the CHT strategy at four budgets (10^2, 10^3, 10^4, 10^5). Each instance and budget combination was then allocated a binary label indicating whether the strategy was one of the best strategies (i.e. a rank of 1) on that instance with that budget.

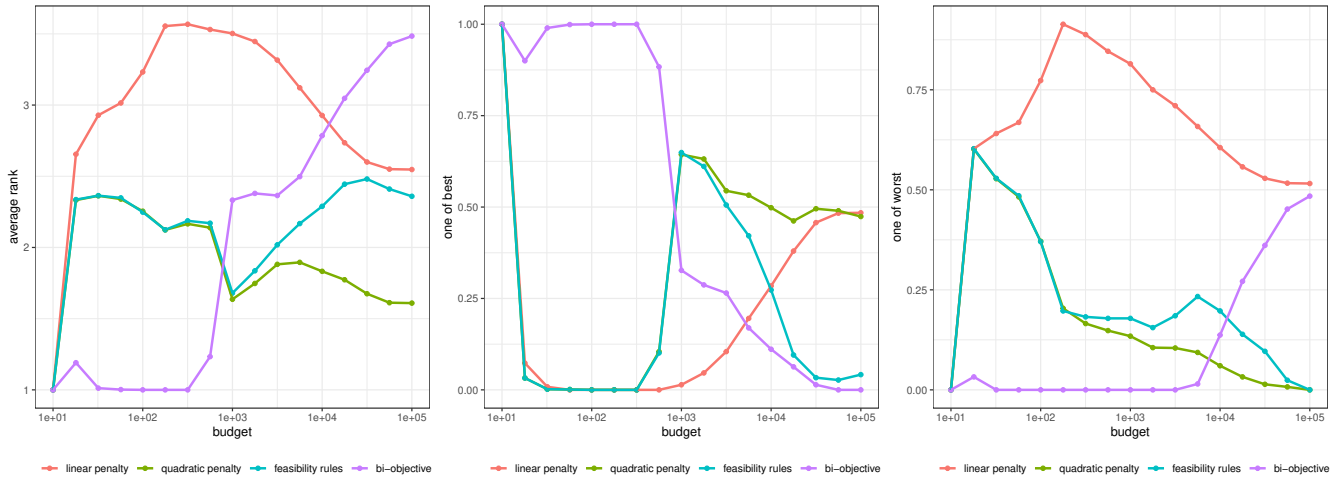

Figure 1: Performance of four constraint handling approaches on 1 080 training instances of the knapsack problem over the budget of function evaluations. Plots from left to right: (1) average rank – lower is better, (2) proportion of instances on which the approach is one of the best, (3) proportion of instances on which the approach is one of the worst.

Table 2: Performance of four constraint handling approaches on 1 080 training instances of the knapsack problem after a budget of 1 000 and 100 000 evaluations.

	Budget: 1 000					Budget: 100 000				
Strategy	Mean rank	Best performing		Worst performing		Mean rank	Best performing		Worst performing	
Linear penalty (LP)	3.50	15	1.4%	880	81.5%	2.55	523	48.4%	557	51.6%
Quadratic penalty (QP)	1.64	695	64.4%	145	13.4%	1.61	512	47.4%	0	0.0%
Feasibility rules (FR)	1.68	701	64.9%	193	17.9%	2.36	45	4.2%	0	0.0%
Bi-objective (BO)	2.33	353	32.7%	0	0.0%	3.48	0	0.0%	523	48.4%

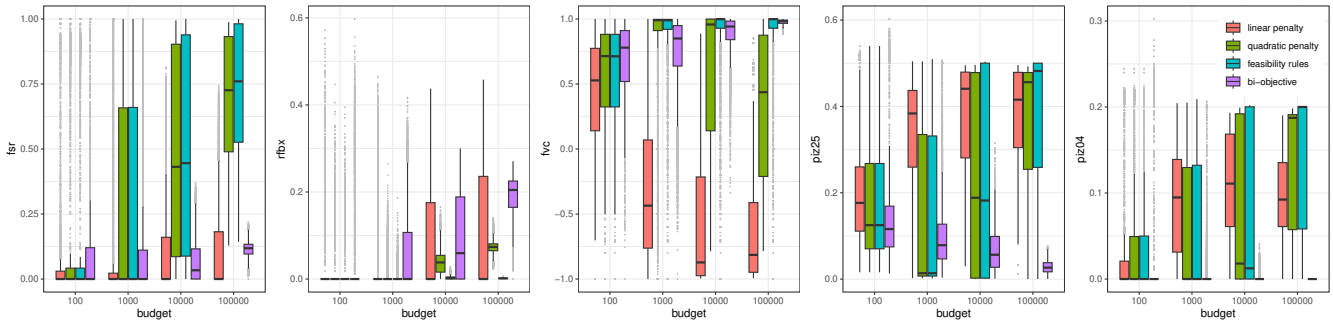

Figure 2: Box plots showing the distribution of landscape features over all training instances for each sampling strategy (evolutionary algorithm with CHT) with increasing budgets.

From these training sets, decision tree models were induced for each CHT strategy where the binary classification task was whether the strategy is one of the best or not. The C4.5 algorithm [23] was used for the training (implemented in the WEKA tool [9] as J48). An example of such a decision tree is provided in Figure 3 for linear penalty (LP). The LP tree model achieved an accuracy of 98.8% on the training data. We see from the tree that the root note is `rfbx`, indicating that this feature was the most important in the prediction task, followed by the budget. At each node, the numbers in brackets give the number of correctly classified instances followed by the number of incorrectly classified instances. For the remaining strategies, the accuracies and root nodes were as follows:

Table 3: Illustrative example of the landscape-aware switching strategy on a single instance of the knapsack problem. For each time step (equivalent to the budget), the five feature-values are reported, followed by the CHTs detected as the best performing ones among which the approach selects one that is run until the next switch.

Time step	fsr	rfbx	fvc	piz25	piz04	Candidate CHTs	Selected CHT
10	0.000	0.000	0.636	0.300	0.000	BO	BO
18	0.000	0.000	0.709	0.176	0.000	BO	BO
32	0.000	0.000	0.744	0.192	0.000	BO	BO
57	0.000	0.000	0.836	0.119	0.000	BO	BO
100	0.000	0.000	0.845	0.081	0.000	BO	BO
178	0.000	0.000	0.849	0.090	0.000	∅	QP
317	0.000	0.000	0.892	0.078	0.000	∅	LP
563	0.000	0.000	0.887	0.060	0.000	∅	BO
1 000	0.000	0.000	0.545	0.228	0.000	∅	FR
1 779	0.000	0.000	0.892	0.061	0.000	QP	QP
3 163	0.000	0.000	0.974	0.004	0.000	QP, FR	FR
5 624	0.173	0.004	0.976	0.001	0.000	QP	QP
10 000	0.356	0.022	0.962	0.106	0.000	LP, QP	QP
17783	0.498	0.041	0.892	0.230	0.086	LP, QP, BO	QP
31 623	0.592	0.052	0.765	0.320	0.172	LP, BO	LP
56 235	0.345	0.031	-0.057	0.197	0.000	LP	LP

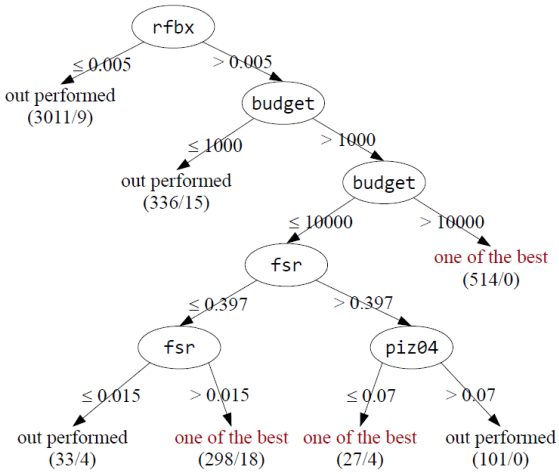

Figure 3: Decision tree for predicting when a linear penalty (LP) strategy will be one of the best strategies.

- Quadratic penalty (QP): accuracy of 94.7% with `fvc` as root node.
- Feasibility rules (FR): accuracy of 95.3% with `piz25` as root node.
- Bi-objective strategy (BO): accuracy of 98.4% with `budget` as root node.

It is interesting that the four decision trees have different features as the root node, which indicates that different features are important for different strategies.

To capture the training knowledge, rules were manually extracted from each of the trees to produce the following.

(1) LP is predicted to be one of the best when:
`rfbx` > 0.005 AND (`budget` > 1 000 AND `budget` ≤ 10 000 AND ((`fsr` ≤ 0.397 AND `fsr` > 0.015) OR (`fsr` > 0.397 AND `piz04` ≤ 0.070)) OR (`budget` > 10 000)).
(2) QP is predicted to be one of the best when:
(`fvc` ≤ 0.911 AND ((`rfbx` ≤ 0.070 AND `fvc` > 0.790 AND `budget` > 1 000) OR (`rfbx` > 0.070 AND `fsr` ≤ 0.776 AND `budget` > 10 000 AND `piz25` ≤ 0.470))) OR (`fvc` > 0.911 AND `budget` > 100 AND (`rfbx` ≤ 0.001 OR (`rfbx` > 0.001 AND ((`budget` ≤ 1 000 AND `rfbx` ≤ 0.020) OR (`budget` > 1 000))))).
(3) FR is predicted to be one of the best when:
`piz25` ≤ 0.030 AND `fsr` ≤ 0.018.
(4) BO is predicted to be one of the best when:
`budget` ≤ 100 OR `budget` > 100 AND ((`piz04` ≤ 0.001 AND `budget` ≤ 1 000 AND `fsr` > 0.0001) OR (`piz04` > 0.001 AND `rfbx` > 0.001 AND ((`fsr` > 0.130) OR (`fsr` ≤ 0.130 AND `budget` ≤ 1 000)))).

The rules above were coded into the landscape-aware approach and used as the basis for choosing appropriate CHTs during the search process, as described below.

5.2 Switching Strategy

The landscape-aware switching strategy was implemented as follows. The base algorithm starts by using a bi-objective CHT. The CHT is switched at set intervals based on the number of function evaluations as defined in Section 3.3. At each switch, the landscape features are calculated using the archive of solutions from the search history (see Section 4.2). Using these features and the current number of function evaluations (budget), the rules are evaluated to determine the set of strategies that are predicted to be the best performing ones. If more than one CHT is predicted to be the best

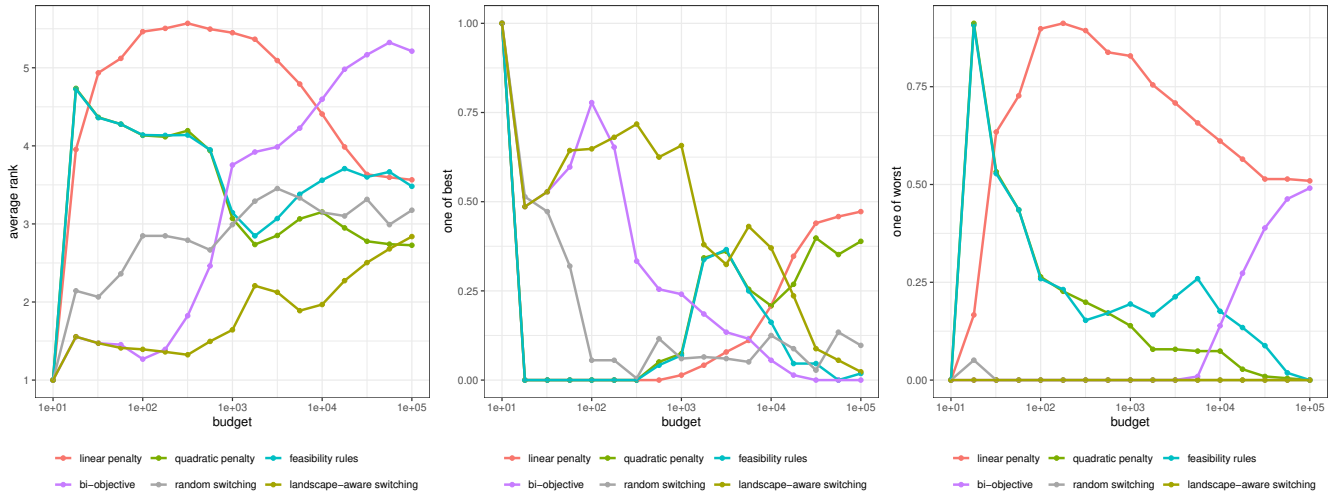

Figure 4: Performance of the four constraint handling approaches, together with the random and landscape-aware switching strategies on 216 test instances of the knapsack problem over the budget of function evaluations. Plots from left to right: (1) average rank, (2) proportion of instances on which the approach is one of the best, (3) proportion of instances on which the approach is one of the worst.

Table 4: Performance of six constraint handling approaches on 216 test instances of the knapsack problem after a budget of 1 000 and 100 000 evaluations.

	Budget: 1 000					Budget: 100 000				
Strategy	Mean rank	Best performing		Worst performing		Mean rank	Best performing		Worst performing	
Linear penalty (LP)	5.45	3	1.4%	179	82.9%	3.56	102	47.2%	110	50.9%
Quadratic penalty (QP)	3.07	16	7.4%	30	13.9%	2.73	84	38.9%	0	0.0%
Feasibility rules (FR)	3.14	15	6.9%	42	19.4%	3.48	4	1.9%	0	0.0%
Bi-objective (BO)	3.75	52	24.1%	0	0.0%	5.21	0	0.0%	106	49.1%
Random switching (RS)	2.99	13	6.0%	0	0.0%	3.18	21	9.7%	0	0.0%
Landscape aware (LA)	1.64	142	65.7%	0	0.0%	2.84	5	2.3%	0	0.0%

performing, then a random choice is made between the candidate CHTs. If no CHT is predicted to be the best performing, then a random choice is made between the four CHTs.

To illustrate this approach, an example run of the switching strategy is given in Table 3 on a sample test instance. The sample run shows that until 100 function evaluations, BO is predicted to be the best strategy, so is selected. For the next four switching events, none of the CHTs were predicted to be the best, so random choices were made. After 1 779 function evaluations, QP is predicted to be the best, and so on.

As an additional baseline, we implement a random switching strategy that switches at the same intervals as the landscape-aware approach, but without using the landscape features and rules.

5.3 Experimental Results

To evaluate the approach, we generate a separate set of test instances following a similar setting than training instances with a number of items of $n = 1\,000$, a profit-weight correlation of $\rho \in \{-0.9, -0.7, -0.4, -0.2, 0.0, 0.2, 0.4, 0.7, 0.9\}$ and a proportional capacity of $c \in \{0.05, 0.10, \ldots, 0.6\}$ with respect to the sum of weights. For each combination of ρ and c, 2 random instances were generated independently of training instances. In total there are thus 216 test instances (9 values of $\rho \times 12$ values of $c \times 2$ random seeds).

The experimental results of the different CHTs and of the switching strategies on the test instances are given in Figure 4. Similar to Section 3.4, the figure shows, from left to right: (1) the average rank, (2) the proportion of times that the considered approach was a best performing one, (3) the proportion of times that the approach was a worst performing one.

Figure 4 shows that the landscape-aware (LA) switching approach has the lowest average rank for most of the budgets. It can also be seen that LA is frequently one of the best strategies and is never one of the worst strategies. Table 4 gives the performance values of the six approaches at budgets of 1 000 and 100 000 function evaluations. The table shows that the LA approach has the lowest

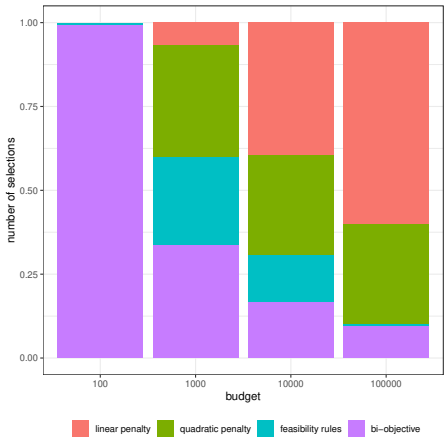

Figure 5: Proportion of times each constraint handling approach is selected by the landscape-aware switching strategy on test instances.

mean rank at budget 1 000 and the second lowest mean rank at budget 100 000 (after QP). The LP approach is the most variable at budget 100 000, performing the best on the most instances (102), but also performing the worst on the most instances (110).

To illustrate how the LA approach adapts to different budget allocations, Figure 5 gives the proportion of times each CHT is selected by the landscape-aware switching strategy. The adaptive strategy starts off by using BO the most and then changes to using LP the most at the highest budget.

6 CONCLUSION

For evolutionary algorithms to be generally applicable to solving real-world problems, effective and adaptive constraint handling approaches are needed. This study showed that there is performance complementarity between different CHTs when used with an evolutionary algorithm to solving instances of the knapsack problem. We demonstrated how an adaptive landscape-aware approach can be implemented using the search trajectory data as a sample for calculating landscape features on-the-fly. By switching to strategies that are predicted to perform the best, the landscape-aware approach out-performed the constituent CHT approaches under most budget scenarios.

Possible future work includes applying landscape-aware constraint handling to solving other combinatorial optimization problems to verify whether the approach is also effective for solving other problem classes. Extensions to the classic knapsack problem would be interesting to investigate, such as the knapsack problem with conflicts [3], where incompatible items cannot be placed together in the knapsack, and the multidimensional knapsack problem [7] with multiple knapsacks and associated constraints, as well as with additional Boolean satisfiability constraints [10]. Another possible benchmark set is the Broken Hill problem [15], which was designed as a tunable test-set for constraint handling approaches. In addition, we are considering extending the popular NK-landscapes [11] to include tunable constraints as a test-bed for constrained optimization with inspiration from the MNK-landscapes of Aguirre and Tanaka [1].

REFERENCES

[1] Hernán Aguirre and Kiyoshi Tanaka. 2007. Working principles, behavior, and performance of MOEAs on MNK-landscapes. *European Journal of Operational Research* 181, 3 (2007), 1670–1690.
[2] BBOB. 2022. GECCO Workshop on Black-Box Optimization Benchmarking (BBOB 2022). https://numbbo.github.io/workshops/BBOB-2022/index.html.
[3] Andrea Bettinelli, Valentina Cacchiani, and Enrico Malaguti. 2017. A Branch-and-Bound Algorithm for the Knapsack Problem with Conflict Graph. *INFORMS Journal on Computing* 29, 3 (2017), 457–473.
[4] Carlos A. Coello Coello. 1999. *A Survey of Constraint Handling Techniques used with Evolutionary Algorithms.* Technical Report. Laboratorio Nacional de Informática Avanzada.
[5] Kalyanmoy Deb. 2000. An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering* 186, 2 (2000), 311 – 338.
[6] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6, 2 (2002), 182–197.
[7] Arnaud Fréville. 2004. The multidimensional 0–1 knapsack problem: An overview. *European Journal of Operational Research* 155, 1 (2004), 1–21.
[8] Guanqiang Gao, Bin Xin, Yi Mei, Shengyu Lu, and Shuxin Ding. 2022. A multi-objective evolutionary algorithm with new reproduction and decomposition mechanisms for the multi-point dynamic aggregation problem. In *Proceedings of the Genetic and Evolutionary Computation Conference.* ACM.
[9] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA Data Mining Software: An Update. *SIGKDD Explor. Newsl.* 11, 1 (2009), 10–18.
[10] Felipe Honjo Ide, Hernan Aguirre, Minami Miyakawa, and Darrell Whitley. 2021. A Generator for Scalable SAT Constrained Multi-Objective Optimization Benchmark Problems. In *2021 IEEE Symposium Series on Computational Intelligence (SSCI).* IEEE.
[11] Stuart A. Kauffman. 1993. *The Origins of Order: Self-Organization and Selection in Evolution.* Oxford University Press, USA.
[12] Pascal Kerschke, Holger H. Hoos, Frank Neumann, and Heike Trautmann. 2019. Automated Algorithm Selection: Survey and Perspectives. *Evolutionary Computation* 27, 1 (March 2019), 3–45.
[13] Katherine M. Malan. 2018. Landscape-Aware Constraint Handling Applied to Differential Evolution. In *Theory and Practice of Natural Computing (Lecture Notes in Computer Science, Vol. 11324).* Springer International Publishing, 176–187.
[14] Katherine M. Malan and Andries P. Engelbrecht. 2013. A survey of techniques for characterising fitness landscapes and some possible ways forward. *Information Sciences* 241 (2013), 148–163.
[15] Katherine M. Malan and I. Moser. 2019. Constraint Handling Guided by Landscape Analysis in Combinatorial and Continuous Search Spaces. *Evolutionary Computation* 27, 2 (2019), 267–289.
[16] K. M. Malan, J. F. Oberholzer, and A. P. Engelbrecht. 2015. Characterising constrained continuous optimisation problems. In *2015 IEEE Congress on Evolutionary Computation (CEC).* 1351–1358.
[17] R. Mallipeddi and P. N. Suganthan. 2010. Ensemble of Constraint Handling Techniques. *IEEE Transactions on Evolutionary Computation* 14, 4 (2010), 561–579.
[18] Rammohan Mallipeddi and Ponnuthurai Nagaratnam Suganthan. 2010. *Problem definitions and evaluation criteria for the CEC 2010 competition on constrained real-parameter optimization.* Technical Report. Nanyang Technological University, Singapore.
[19] Silvano Martello, David Pisinger, and Paolo Toth. 1999. Dynamic Programming and Strong Bounds for the 0-1 Knapsack Problem. *Management Science* 45, 3 (1999), 414–424.
[20] Efrén Mezura-Montes and Carlos A. Coello Coello. 2011. Constraint-handling in nature-inspired numerical optimization: Past, present and future. *Swarm and Evolutionary Computation* 1, 4 (2011), 173–194.
[21] Zbigniew Michalewicz. 1995. A Survey of Constraint Handling Techniques in Evolutionary Computation Methods. *Evolutionary Programming* 4 (1995), 135–155.
[22] Zbigniew Michalewicz and Jarosław Arabas. 1994. Genetic algorithms for the 0/1 knapsack problem. In *Methodologies for Intelligent Systems*, Zbigniew W. Raś and Maria Zemankova (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 134–143.
[23] J. Ross Quinlan. 1993. *C4.5: Programs for Machine Learning.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
[24] P.N. Suganthan. 2010. *Comparison of Results on the 2010 CEC Benchmark Function Set.* Technical Report. Nanyang Technological University, Singapore. http://www3.ntu.edu.sg/home/epnsugan/index_files/CEC10-Const/CEC2010_COMPARISON2.pdf