Machine Learning Assisted Hybrid Genetic Algorithm Applied to the Knapsack Problem with Forfeits

Gabriel Souto [*1] Claudio Miceli [1] Luidi Simonetti [1] Pedro Henrique González [*1]

Abstract

The hybridization of ML techniques with heuristics approaches is promising for difficult problems. This paper aimed to explore this hybridization in an NP-Hard Problem with a proposed *ML-HBRKGA*, a hybrid approach combining a Biased Random-Key Genetic Algorithm (BRKGA) with Q-learning and a Random Forest Regressor with Local Branching to solve the Knapsack Problem with Forfeits (KPF). The analysis shows that the approach is competitive with the existing literature.

1. Introduction

Managing conflicting or mutually exclusive choices poses a significant challenge in many optimization scenarios. Disjunctive constraints or conflicts have been incorporated into combinatorial optimization problems, enriching classical problems like the Minimum Spanning Tree (Carrabs et al., 2021), Maximum Flow (Galbiati, 2011), Knapsack (Bettinelli et al., 2017; Fügenschuh et al., 2019; Hifi & Otmani, 2012), and Bin Packing (Epstein & Levin, 2008) Problems to address this. This article concentrates on the Knapsack Problem with Forfeits (KPF) (Cerulli et al., 2020), a variation of the 0-1 Knapsack Problem that introduces ”soft” conflict constraints, also called forfeits, which impose a penalty when items from a forfeit pair, are selected together to compose a solution. This change is advantageous when strictly avoiding conflicts leads to infeasible solutions or the cost of preventing conflicts is too high.

The KPF has diverse applications across various fields, including workforce assignment, employee shift scheduling, healthcare, and logistics. For instance, in workforce assignment, specific machines might need to be operated by certain workers, while in shift scheduling, some shifts incur additional costs when assigned together. In healthcare, optimizing food and drug intake to avoid adverse interactions is crucial, and in logistics, ensuring the safe transportation of items requiring special treatment is necessary. These examples highlight the practical importance and flexibility of addressing forfeits in practice (Capobianco et al., 2022).

This paper applies an approach called *ML-HBRKGA* to solve the Knapsack Problem with Forfeits (KPF) using a regression task. This methodology integrates a Biased Random-Key Genetic Algorithm (BRKGA), a Local Branching technique, Q-learning as a reinforcement learning algorithm, and a Random Forest Regressor for the regression task. The *ML-HBRKGA* approach effectively reduces the solution gap and improves computational efficiency compared to existing methods, offering suitable solutions for the KPF.

The paper is organized as follows: Section 2 reviews related literature, and Section 3 provides a detailed description of KPF and a mixed integer programming formulation. Section 4 outlines the proposed method and its components. Section 5 presents the computational experiments and their results. Finally, Section 6 concludes the paper and discusses potential future work.

2. Related Literature

Using the *Scopus* database, the literature review explores the Knapsack Problem with Forfeits (KPF) by focusing on peer-reviewed articles. The initial work on KPF by (Cerulli et al., 2020) introduced a Mixed Integer Programming (MIP) formulation and two heuristics: *GreedyForfeits* and *CarouselForfeits*. *GreedyForfeits* sequentially inserts items into the knapsack based on their benefit-to-weight ratio, while *CarouselForfeits*, inspired by the *Carousel Greedy* paradigm (Cerrone et al., 2017), iteratively improves the solution by re-evaluating early greedy choices.

Subsequent research introduced *ILSForfeits* (Moura et al., 2021), an Iterated Local Search-based heuristic, and *GA-CG* (Capobianco et al., 2022), a hybrid algorithm combining a Genetic Algorithm with *Carousel Greedy*, which demonstrated superior performance in benchmark instances. The success of *GA-CG* led to the exploration of genetic

[*]Equal contribution [1]Federal University of Rio de Janeiro, Rio de Janeiro, Brazil. Correspondence to: Gabriel Souto <soutog@cos.ufrj.br>, Pedro Henrique González <pegonzalez@cos.ufrj.br>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

algorithms and hybrid methods, inspiring the present work, which will be detailed in the following sections.

3. Problem Description

To formally define the problem, we consider the formulation by (Cerulli et al., 2020). Assume a set of items I with n elements. Each item $i \in I$ is characterized by a profit $p_i > 0$ and a weight $w_i > 0$. The total weight of selected items must not exceed a capacity limit $b > 0$. Additionally, we define a set K containing $|K|$ forfeit pairs, where including both items of a pair incurs a penalty.

Each element in $k = \{i, j\} \in K$ consists of a pair of elements, where $i, j \in I$, with a corresponding forfeit cost $d_k > 0, k = 1, 2, ...l$. So, the Mixed Integer Programming model (MIP) for KPF can be described as:

$$\max \sum_{i \in \mathcal{I}} p_i x_i - \sum_{k \in \mathcal{K}} d_k v_k$$

$$\sum_{i \in \mathcal{I}} w_i x_i \leq b \tag{1}$$

$$x_i + x_j - v_k \leq 1 \qquad \forall k = \{i, j\} \in \mathcal{K} \tag{2}$$

$$x_i \in \{0, 1\} \qquad \forall i \in I \tag{3}$$

$$0 \leq v_k \leq 1 \qquad \forall k \in \mathcal{K} \tag{4}$$

The MIP includes binary variables x_i, representing whether each item i from set I is included in the knapsack. A variable v_k is also introduced to indicate if the forfeit cost d_k associated with a forfeit pair must be paid. Constraint (1) guarantees that the total weight of selected items does not exceed the knapsack's maximum capacity. Constraints (2) implement conditions on the variables x and v, specifying that v_k must be set to 1 when both items $k = \{i, j\}$ are selected. The objective function focuses on maximizing profit while minimizing forfeit costs.

4. Methodology

This section introduces the ML-HBRKGA method, which combines Machine Learning Algorithms with a Hybrid Biased Random-Key Genetic Algorithm to address the KPF. The *ML-HBRKGA* includes several components: a constructive heuristic for generating the initial population, local search during the decoding phase, a Q-learning algorithm for deciding when to apply local search, and a Local Branching technique to enhance solution quality with a regression task that uses a Random Forest (RF). The workflow of the ML-HBRKGA method is illustrated in Figure 1, with detailed explanations of each component in the subsequent subsections.

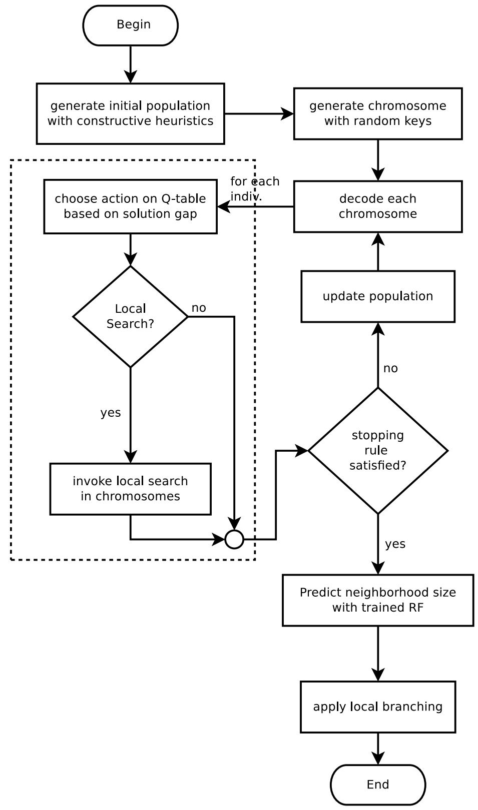

Figure 1. ML-HBRKGA Flowchart

4.1. BRKGA

At the core of our approach lies the Biased Random Key Genetic Algorithm (BRKGA), introduced by (Gonçalves & Resende, 2011). Unlike its predecessor, RKGA (Bean, 1994), BRKGA offers a unique parent selection for crossover. Operating with real-valued chromosomes within the range [0,1], BRKGA employs problem-specific decoders to transform these values into solutions and compute their objective functions. The algorithm divides solutions into elite and non-elite categories, with a dedicated elitism procedure creating a separate population. New members are generated through crossover (mixing elite and non-elite solutions) and mutation (introducing random changes to specific keys).

BRKGA's bias prefers elite solutions during genetic information transmission associated with crossover, confirming its effectiveness in solving many combinatorial optimization problems. These include Facility Location Problems (Souto et al., 2021; Morais et al., 2022), Traveling Salesman Problem Variants (Snyder & Daskin, 2006; Samanlioglu et al., 2008), Job Shop Scheduling (Brandao et al., 2015; Brandão et al., 2017) and many others.

An essential phase of BRKGA is the decoder phase. The decoder algorithm takes a chromosome of random keys as input and computes the fitness of the obtained solution. In the implemented decoder for the KPF, a binary vector represents a solution, which is decoded by sorting the keys and selecting items until the knapsack capacity is reached. The algorithm then extracts the state and action of the current solution from the trained Q-table by the Q-learning algorithm to decide whether to apply the Local Search or not.

4.2. Constructive Heuristic

The Constructive Heuristic is crucial in creating the initial population for BRKGA. In this article, we utilize a Semi-Greedy approach called *Semi-GreedyForfeits*, inspired by the *GreedyForfeits* method introduced by (Cerulli et al., 2020).

The adaptation involves selecting items from the subset $X_{iter} = \{i | w_i \leq b_{res} \ \wedge \ i \in I \setminus S\}$, which contains items that can still be added to the solution set S. This is done by looping through a Restricted Candidate List (RCL):

$$RCL = \{i \in I | h_{min} \leq h_i \leq h_{max} + \alpha(h_{min} - h_{max})\}, \tag{5}$$

where h is the cost-benefit function and α determines the level of greediness or randomness in the algorithm's item selection process.

4.3. Local Search

In summary, the local search algorithm explores a solution neighborhood to improve upon the current solution iteratively. It focuses on item exchange within the Knapsack Problem with Forfeits (KPF), where $\bar{S}$ represents items not included in the current solution S. By swapping items between $\bar{S}$ and S, the algorithm evaluates if such exchanges enhance the solution's cost. To mitigate computational costs as the instance size increases, we use an efficient solution evaluation process denoted as δ:

$$\delta_{ij} = \beta_j - \beta_i + d_{ij}, \quad \forall i \in S, j \in \bar{S} \tag{6}$$

$$\beta_m = p_m - \sum_{j \in \mathcal{S}} d_{mj}, \quad \forall m \in I \tag{7}$$

In the worst-case scenario, evaluating the solution has a time complexity of $\mathcal{O}(1)$. Using this approach reduces the local search worst-case complexity from $\mathcal{O}(n^4)$ to $\mathcal{O}(n^2)$.

4.4. Machine Learning Stage: Q-Learning and Random Forest

In Machine Learning, Reinforcement Learning (RL) involves an agent interacting with an environment by taking actions, receiving rewards, and transitioning to new states to find optimal solutions. This approach models problems as sequential decision-making processes using the *Markov Decision Process (MDP)* framework (Mazyavkina et al., 2021; Sutton & Barto, 2018; Chaves & Lorena, 2021; Bellman, 1957).

In the Q-learning process, a Q-Table is iteratively updated using an epsilon-greedy policy to guide the agent's actions, aiming to maximize long-term cumulative rewards (Watkins & Dayan, 1992; Watkins, 1989). In the context of KPF, the Q-learning algorithm is integrated into the BRKGA decoder phase to decide whether to apply local search based on the gap value, calculated as $gap = 1 - (S_{best}/S_{current})$. Here, S_{best} is the best solution, and $S_{current}$ is the current solution. The Q-Table is initialized with all values set to 0, indicating the starting point and the decision to apply local search. The state is defined by the difference between the current solution cost and the cost of the best solution found. The action is determined by selecting the maximum value from the trained Q-table that corresponds to the given state

Another crucial step in the implemented learning process in Local Branching (LB) was inspired in the works (Fischetti & Lodi, 2003; Liu et al., 2021). The Local Branching (LB) technique serves to refine existing feasible solutions. Like the local search process, LB incorporates linear inequalities into the model, thereby defining neighborhoods for exploration (Gonzalez et al., 2016; Gonzalez & Brandão, 2018).

Considering the KPF, let us examine a solution $s \in P$, where P delineates the polyhedron defined by Constraints (1) - (4). Implementing this approach involves incorporating the following LB constraint:

$$\sum_{i \in I | \bar{x}_i = 0} x_i + \sum_{i \in I | \bar{x}_i = 1} (1 - x_i) \leq \Delta, \tag{8}$$

where Δ is a given positive integer, indicating the number of variables x_i, $i \in I$, which can change their value from one to zero and vice versa.

In the learning phase, a Random Forest (Wang & Jin, 2018; Liaw et al., 2002) was applied to predict a precise neighborhood size for each instance of the KPF.

5. Computational Experiments

This section contains two subsections. The first one summarizes the implementation details and tuning parameters, while the remaining examines the key findings related to *ML-HBRKGA*, providing insightful comparisons with existing literature.

5.1. Implementation Details and Parameters Tuning

The proposed method, *ML-HBRKGA*, was implemented in Python using the Python interpreter version 3.10.12 and IBM CPLEX version 22.1.1. The experiments were executed three times for each instance on an AMD Ryzen 5 4600G with a Radeon Graphics processor running at 4.30 GHz and 16 GB RAM.

Before experimentation, we first tuned the parameters for *ML-HBRKGA* using *iRace* (López-Ibáñez et al., 2016) to optimize the Hybrid BRKGA (BRKGA + LB) settings, including the α value for the Restricted Candidate List (RCL) in *Semi-GreedyForfeits*, and BRKGA parameters: population size (p), elite population size (p_e) and mutation ratio (p_m). In the second phase, we used *Grid-Search* to fine-tune Q-learning parameters such as learning rate (κ), number of training episodes (n_{ep}), maximum Q-learning steps (max_{steps}), discount rate (γ), decay rate (r), and exploration/exploitation probabilities ($\epsilon_{min} = 0.01$, $\epsilon_{max} = 1.0$). These parameters were finalized after extensive testing, and the algorithm was set to stop after 180 seconds.

For the experiments, we used the ”O” (Original) set of ten instances described in (Cerulli et al., 2020), which included 500, 700, 800, or 1000 items ($n = |I|$). Each set also had corresponding forfeit pairs ($l = 6n$) and capacity ($b = 3n$). The weights, profits, and forfeit costs for each item were randomly generated from the intervals [3, 20], [5, 25], and [2, 15], respectively. It's also significant to mention that the gap was calculated in comparison with the best solution found using CPLEX with a time limit of 7200s, as described in (Capobianco et al., 2022).

5.2. ML-HBRKGA versus literature

After tuning the algorithm, experiments were conducted to compare the results of the proposed methodology with those of the literature. An important metric used for this comparison was the solution gap, defined as $(1 - (Sol_{heuristic}/Sol_{CPLEX})) \times 100\%$, where $Sol_{heuristic}$ is the cost of the heuristic solution, and Sol_{CPLEX} is the cost of the best solution obtained through IBM CPLEX solver.

The proposed methodology was compared with leading KPF approaches, including *CarouselForfeits* and *GA-CG*. *GA-CG* was selected over *ILSForfeits* since it performs better, and because *ILSForfeits* did not report solution values for all instance classes, it was excluded from the analysis.

Table 1 indicates in bold that the mean gap of *ML-HBRKGA* is superior to other methods in the literature, and the execution time follows the same tendency. Importantly, even as the number of items increases, *ML-HBRKGA* consistently finds high-quality solutions with a gap of less than 1%.

The results showed that the proposed *ML-HBRKGA* is a viable approach to solve the KPF since, in the worst case, the method was able to find solutions that were at most 2.32% away from the best known lower bound at a time close to 3 minutes. From an application point of view, finding better solutions quickly and with a smaller distance from the optimal solution seems promising for practical situations.

Table 1. Mean GAP and time for “O” set of instances

Metrics	Methods	Number of items			
		500	700	800	1000
Mean Gap(%)	CarouselForfeits	4.39	5.18	4.61	23.03
	GA-CG	1.53	1.70	1.60	1.82
	ML-HBRKGA	**0.02**	**0.20**	**0.52**	**0.47**
Mean Time(s)	CarouselForfeits	1.35	3.61	5.53	5.53
	GA-CG	163.90	506.22	798.20	1592.94
	ML-HBRKGA	177.37	216.51	233.06	265,63

6. Conclusions and Future Works

The paper addresses the Knapsack Problem with Forfeits (KPF), a modified version of the classic 0-1 Knapsack Problem. In KPF, the complexity arises from introducing forfeit pairs, called soft conflicts, which negatively impact the solution's total cost. To solve this NP-hard problem, we propose ML-HBRKGA. This innovative approach integrates Q-Learning with a Hybrid Biased Random-Key Genetic Algorithm and Random Forest to predict the neighborhood size of Local Branching constraints. Our method proposes contributing new perspectives and solutions to this challenging problem.

Our implementation integrates a Hybrid Biased Random-Key Genetic Algorithm (HBRKGA) with Q-learning, featuring a ”smart” decoder trained with Q-learning to determine the application of a local search algorithm dynamically. Additionally, we incorporated a final Local Branching technique, coupled with Random Forest, to predict the optimal neighborhood size for each instance and enhance the solution's feasibility.

The experimental findings highlight the efficacy of the ML-HBRKGA approach in solving the KPF and generating high-quality solutions. Analysis of the solution gaps indicates that the ML-HBRKGA method generates solutions with gaps smaller than 1%, emphasizing its reliability and efficiency.

At last, the Knapsack Problem with Forfeits using ML-HBRKGA is just the beginning of the research. While these initial findings are promising. Further investigation into the application of Machine Learning techniques, both exact and heuristic, is essential. This approach is not limited to KPF but extends to other variations of the knapsack problem, including those with hard item conflicts and sets of soft conflicts. These explorations could yield valuable advancements in the field.

References

Bean, J. C. Genetic algorithms and random keys for sequencing and optimization. *ORSA journal on computing*, 6(2):154–160, 1994.

Bellman, R. A markovian decision process. *Journal of mathematics and mechanics*, pp. 679–684, 1957.

Bettinelli, A., Cacchiani, V., and Malaguti, E. A branch-and-bound algorithm for the knapsack problem with conflict graph. *INFORMS Journal on Computing*, 29(3):457–473, 2017.

Brandao, J. S., Noronha, T. F., Resende, M. G., and Ribeiro, C. C. A biased random-key genetic algorithm for single-round divisible load scheduling. *International Transactions in Operational Research*, 22(5):823–839, 2015.

Brandão, J. S., Noronha, T. F., Resende, M. G., and Ribeiro, C. C. A biased random-key genetic algorithm for scheduling heterogeneous multi-round systems. *International Transactions in Operational Research*, 24(5):1061–1077, 2017.

Capobianco, G., D'Ambrosio, C., Pavone, L., Raiconi, A., Vitale, G., and Sebastiano, F. A hybrid metaheuristic for the knapsack problem with forfeits. *Soft Computing*, pp. 1–14, 2022.

Carrabs, F., Cerulli, R., Pentangelo, R., and Raiconi, A. Minimum spanning tree with conflicting edge pairs: a branch-and-cut approach. *Annals of Operations Research*, 298(1-2):65–78, 2021.

Cerrone, C., Cerulli, R., and Golden, B. Carousel greedy: A generalized greedy algorithm with applications in optimization. *Computers & Operations Research*, 85:97–112, 2017.

Cerulli, R., D'Ambrosio, C., Raiconi, A., and Vitale, G. The knapsack problem with forfeits. In *Combinatorial Optimization: 6th International Symposium, ISCO 2020, Montreal, QC, Canada, May 4–6, 2020, Revised Selected Papers 6*, pp. 263–272. Springer, 2020.

Chaves, A. A. and Lorena, L. H. N. An adaptive and near parameter-free brkga using q-learning method. In *2021 IEEE Congress on Evolutionary Computation (CEC)*, pp. 2331–2338. IEEE, 2021.

Epstein, L. and Levin, A. On bin packing with conflicts. *SIAM Journal on Optimization*, 19(3):1270–1298, 2008.

Fischetti, M. and Lodi, A. Local branching. *Mathematical Programming*, 98:23–47, 2003.

Fügenschuh, A., Bambach, M., and Buhl, J. Trajectory optimization for wire-arc additive manufacturing. In *Operations Research Proceedings 2018: Selected Papers of the Annual International Conference of the German Operations Research Society (GOR), Brussels, Belgium, September 12-14, 2018*, pp. 331–337. Springer, 2019.

Galbiati, G. Approximating minimum cut with bounded size. In *International Conference on Network Optimization*, pp. 210–215. Springer, 2011.

Gonçalves, J. F. and Resende, M. G. Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics*, 17(5):487–525, 2011.

Gonzalez, P. H. and Brandão, J. A biased random key genetic algorithm to solve the transmission expansion planning problem with re-design. In *2018 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–7. IEEE, 2018.

Gonzalez, P. H., Simonetti, L., Michelon, P., Martinhon, C., and Santos, E. A variable fixing heuristic with local branching for the fixed charge uncapacitated network design problem with user-optimal flow. *Computers & Operations Research*, 76:134–146, 2016.

Hifi, M. and Otmani, N. An algorithm for the disjunctively constrained knapsack problem. *International Journal of Operational Research*, 13(1):22–43, 2012.

Liaw, A., Wiener, M., et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

Liu, D., Fischetti, M., and Lodi, A. Revisiting local branching with a machine learning lens. *arXiv preprint arXiv:2112.02195*, 2021.

López-Ibáñez, M., Dubois-Lacoste, J., Cáceres, L. P., Birattari, M., and Stützle, T. The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3:43–58, 2016.

Mazyavkina, N., Sviridov, S., Ivanov, S., and Burnaev, E. Reinforcement learning for combinatorial optimization: A survey. *Computers & Operations Research*, 134: 105400, 2021.

Morais, I., Souto, G., Ribeiro, G. M., Mendonça, I., and González, P. H. A hybrid brkga approach for the multi-product two stage capacitated facility location problem. In *2022 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8. IEEE, 2022.

Moura, A. F. C., Noronha, T., and Bogue, E. T. Uma heurística ils para o problema da mochila com penalidades. *Simpósio Brasileiro de Pesquisa Operacional (SBPO)*, 2021.

Samanlioglu, F., Ferrell Jr, W. G., and Kurz, M. E. A memetic random-key genetic algorithm for a symmetric multi-objective traveling salesman problem. *Computers & Industrial Engineering*, 55(2):439–449, 2008.

Snyder, L. V. and Daskin, M. S. A random-key genetic algorithm for the generalized traveling salesman problem. *European journal of operational research*, 174(1):38–53, 2006.

Souto, G., Morais, I., Faulhaber, L., Ribeiro, G. M., and González, P. H. A hybrid brkga approach for the two stage capacitated facility location problem. In *2021 IEEE Congress on Evolutionary Computation (CEC)*, pp. 2007–2014. IEEE, 2021.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Wang, H. and Jin, Y. A random forest-assisted evolutionary algorithm for data-driven constrained multiobjective combinatorial optimization of trauma systems. *IEEE transactions on cybernetics*, 50(2):536–549, 2018.

Watkins, C. J. and Dayan, P. Q-learning. *Machine learning*, 8:279–292, 1992.

Watkins, C. J. C. H. Learning from delayed rewards. *Machine learning*, 1989.