The Journal of Open Source Software

QMKPy: A Python Testbed for the Quadratic Multiple Knapsack Problem

Karl-Ludwig Besser [1] **and Eduard A. Jorswieck** [1]

1 Institute for Communications Technology, Technische Universität Braunschweig, Germany

DOI: 10.21105/joss.04718

Software

- Review
- Repository
- Archive

Editor: Mehmet Hakan Satman

Reviewers:

- @ulf1
- @max-little

Submitted: 08 August 2022
Published: 02 November 2022

Summary

QMKPy provides a Python framework for modeling and solving the quadratic multiple knapsack problem (QMKP). It is primarily aimed at researchers who develop new solution algorithms for the QMKP. QMKPy therefore mostly functions as a testbed to quickly implement novel algorithms and compare their results with existing ones. However, the package also already includes implementations of established algorithms for those who only need to solve a QMKP as part of their research.

The QMKP is a type of knapsack problem which has first been analyzed by Hiley & Julstrom (2006). For a basic overview of other types of knapsack problems see, e.g., Kellerer et al. (2004). As in the regular multiple knapsack problem, the goal is to assign $N \in \mathbb{N}$ items with given weights $w_i \in \mathbb{R}_+$ and (non-negative) profit values $p_i \in \mathbb{R}_+$ to $K \in \mathbb{N}$ knapsacks with given weight capacities $c_u \in \mathbb{R}_+$, such that a total profit is maximized. In the QMKP, there exist additional joint profits $p_{ij} \in \mathbb{R}_+$ which are yielded when items i and j are packed into the same knapsack.

Mathematically, the QMKP is described by the following optimization problem

$$\max \quad \sum_{u\in\mathcal{K}} \left(\sum_{i\in\mathcal{A}_u} p_i + \sum_{\substack{j\in\mathcal{A}_u\\ j\neq i}} p_{ij}\right) \tag{1a}$$

$$\text{s.t.} \quad \sum_{i\in\mathcal{A}_u} w_i \leq c_u \qquad \forall u \in \mathcal{K} \tag{1b}$$

$$\sum_{u=1}^{K} a_{iu} \leq 1 \qquad \forall i \in \{1, 2, \dots, N\} \tag{1c}$$

where $\mathcal{K} = \{1, 2, \dots, K\}$ describes the set of knapsacks, $\mathcal{A}_u \subseteq \{1, 2, \dots, N\}$ is the set of items that are assigned to knapsack u and $a_{iu} \in \{0, 1\}$ is a binary variable indicating whether item i is assigned to knapsack u.

Statement of need

The QMKP is an NP-hard optimization problem and therefore, there exists a variety of (heuristic) algorithms to find good solutions for it. While Python frameworks already exist for the standard (multiple) knapsack problem and the quadratic knapsack problem, they do not consider the *quadratic multiple* knapsack problem. However, this type of problem arises in many areas of research. In addition to the typical problems in Operations Research, it also occurs in distributed computing (Rust et al., 2020) and in the area of wireless communications (Besser et al., 2022).

For the classic knapsack problem and the quadratic *single* knapsack problem, many well-known optimization frameworks like Gurobi (Gurobi Optimization, LLC, 2022) and OR-Tools (Perron & Furnon, 2022) provide solution routines. However, they are not directly applicable to the QMKP. Furthermore, it can be difficult for researchers to reproduce results that rely on commerical software.

Therefore, QMKPy aims to close that gap by providing an open source testbed to easily implement and compare solution algorithms. Additionally, Python is widely used among researchers and enables easy to read implementations. This further supports the goal of QMKPy to promote sharable and reproducible solution algorithms.

For initial comparisons, the software already implements multiple solution algorithms for the QMKP, including a *constructive procedure (CP)* based on Algorithm 1 from Aïder et al. (2022) and the greedy heuristic from Hiley & Julstrom (2006). A second algorithm that is included is the *fix and complete solution (FCS) procedure* from Algorithm 2 in Aïder et al. (2022). Additionally, a collection of reference QMKP instances that can be used with QMKPy is provided at (Besser, 2022). This dataset includes the well-known reference problems from Hiley & Julstrom (2006), which in turn are based on the quadratic single knapsack problems from Billionnet & Soutif (2004).

The open source nature of QMKPy and its aim at researchers encourages the implementation of more algorithms for solving the QMKP that can become part of the QMKPy framework.

The most notable benefits when implementing an algorithm using QMKPy are the following:

- No additional overhead is required. Only a single function with the novel solution algorithm needs to be implemented.
- Generic unit tests are available to make sure that the novel algorithm fulfills the set of basic criteria.
- The ability of loading and saving problem instances allows for quick and easy testing of any algorithm against reference datasets. This enables reproducible research and creates a high degree of comparability between different algorithms.

References

Aïder, M., Gacem, O., & Hifi, M. (2022). Branch and solve strategies-based algorithm for the quadratic multiple knapsack problem. *Journal of the Operational Research Society*, *73*(3), 540–557. https://doi.org/10.1080/01605682.2020.1843982

Besser, K.-L. (2022). *QMKPy datasets* (Version v1.0) [Computer software]. Zenodo. https://doi.org/10.5281/ZENODO.7157144

Besser, K.-L., Jorswieck, E. A., & Coon, J. P. (2022, September). Multi-user frequency assignment for ultra-reliable mmWave two-ray channels. *20th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*.

Billionnet, A., & Soutif, Éric. (2004). Using a mixed integer programming tool for solving the 0–1 quadratic knapsack problem. *INFORMS Journal on Computing*, *16*(2), 188–197. https://doi.org/10.1287/ijoc.1030.0029

Gurobi Optimization, LLC. (2022). *Gurobi Optimizer Reference Manual*. https://www.gurobi.com

Hiley, A., & Julstrom, B. A. (2006). The quadratic multiple knapsack problem and three heuristic approaches to it. *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation - GECCO '06*, 547–552. https://doi.org/10.1145/1143997.1144096

Kellerer, H., Pferschy, U., & Pisinger, D. (2004). *Knapsack problems*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-24777-7

Perron, L., & Furnon, V. (2022). *OR-tools* (Version v9.4) [Computer software]. Google. https://developers.google.com/optimization/

Rust, P., Picard, G., & Ramparany, F. (2020). Resilient distributed constraint optimization in physical multi-agent systems. *24th European Conference on Artificial Intelligence (ECAI 2020)*, 195–202. https://doi.org/10.3233/FAIA200093