

Random-key Optimizers

Antonio A. Chaves ^a

antonio.chaves@unifesp.br

Mauricio G. C. Resende ^{a,b}

mgcr@uw.edu

Ricardo M. A. Silva ^c

rmasc@cin.ufpe.br

^a Universidade Federal de São Paulo - UNIFESP/ICT
São José dos Campos, SP, Brazil

^b Industrial and Systems Engineering, University of Washington
Seattle, WA, U.S.A.

^c Centro de Informática, Universidade Federal de Pernambuco
Recife, PE, Brazil

ABSTRACT

This mini-course presents problem-independent metaheuristics using the random-key optimizer (RKO) paradigm. SA (simulated annealing), GRASP (greedy randomized adaptive search procedure), VNS (variable neighborhood search), ILS (iterated local search), and PSO (particle swarm optimization) are classic metaheuristics for combinatorial optimization. A random-key optimizer (RKO) uses a vector of random keys to encode a solution to a combinatorial optimization problem. It uses a decoder to evaluate a solution encoded by the vector of random keys. An RKO is a metaheuristic where points in the unit hypercube are evaluated using a decoder. We describe RKO as comprising a problem-independent component and a problem-dependent decoder. As a proof of concept, the RKO with different metaheuristics is tested on five NP-hard combinatorial optimization problems: traveling salesman problem, tree of hubs location problem, Steiner triple covering problem, node capacitated graph partitioning problem, and job sequencing and tool switching problem.

KEYWORDS. Metaheuristics. Random key. Optimization problems.

1. Introduction

This course covers problem-independent metaheuristics using the random-key optimizer (RKO) paradigm. An RKO [Schuetz et al., 2022] uses a vector of random keys to encode a solution to a combinatorial optimization problem. It uses a decoder to evaluate a solution encoded by the vector of random keys. A random-key metaheuristic is a metaheuristic where points in the unit hypercube are evaluated using a decoder. We describe a random key metaheuristic consisting of a problem-independent component and a problem-dependent decoder. As a proof of concept, six variants of the random-key metaheuristic are tested on five NP-hard combinatorial optimization problems: traveling salesman problem, tree of hubs location problem, Steiner triple covering problem, node capacitated graph partitioning problem, and job sequencing and tool switching problem. BRKGA (biased random-key genetic algorithm), SA (simulated annealing), GRASP (greedy randomized adaptive search procedure), VNS (variable neighborhood search), ILS (iterated local search), and PSO (particle swarm optimization) are the classic metaheuristics for combinatorial optimization that are used in the RKO.

2. Random-Key Optimizer

Random-key genetic algorithms (RKGA) were first introduced by Bean [1994]. In a RKGA, solutions are encoded as vectors of random keys, i.e., randomly-generated real numbers in the interval $(0, 1]$. A population of p random-key vectors is evolved over a number of generations. The initial population consists of randomly generated n -vectors. At each iteration, each random-key vector is decoded and evaluated. The *decoder* is a deterministic algorithm, usually a heuristic, that takes as input a vector of random keys and returns a feasible solution of the problem being solved along the cost of the solution. Biased random-key genetic algorithms [Gonçalves and Resende, 2011] are an extension of RKGA where a bias is applied in the selection of one of the parents as well as in the mating process.

Both RKGA and BRKGA are problem-independent algorithms in the sense that there is a clear separation between the solver and the problem being solved where the connection of the solver with the problem is done by way of a decoder. For each type of problem to be solved, a new decoder is implemented to serve as the link between the solver and the problem. These algorithms are examples of *Random-Key Optimizers* (RKO) [Schuetz et al., 2022]. An RKO is an optimization heuristic algorithm that solves a discrete optimization problem indirectly in the continuous unit n -dimensional hypercube $\mathcal{H}_n$. For each solution $x \in \mathcal{H}_n$, a decoder $\mathcal{D}$ maps x to the solution $\mathcal{D}(x)$ in the solution space of the discrete optimization problem. With such a separation of solver and decoder, one needs only to implement the solver once and then it can be reused to solve a number of different problems by simply devising a decoder for that problem. Examples of APIs for BRKGA are Toso and Resende [2015]; Andrade et al. [2021]; Oliveira et al. [2022].

Figure 1 illustrates our approach, presenting a comprehensive end-to-end optimization pipeline. In panel (a), we have a schematic depiction of the workflow, commencing with the problem input, characterized by attributes of the problem at hand. This input is then directed into the random-key optimizer (RKO), which orchestrates a heuristic exploration to search for quality solutions, constituting the final output of the pipeline. Expansions to this core routine are feasible by integrating additional metaheuristics, thereby facilitating further refinement of solutions. Analogous to ensemble techniques prevalent in machine learning, amalgamating a diverse pool of solutions can enhance the identification of high-quality solutions. Therefore, solution refinement is attainable through the path-relinking method.

Panel (b) explains the RKO's architecture. Noteworthy is the discernible demarcation between problem-independent modules, displayed by the blue hypercube housing random-key solutions, and the problem-dependent, deterministic decoder tasked with mapping a random-key solution into a solution to the problem at hand, accompanied by an associated cost or fitness value.

The decoder is developed to ensure adherence to problem constraints, such as the mandatory visitation of each node exactly once in a Traveling Salesman Problem (TSP). While the trajectory of a random-key solution in BRKGA adheres to evolutionary principles, facile adaptations to trajectory algorithmic paradigms, such as GRASP, SA, ILS, and VNS, are feasible.

Panel (c) gives an illustrative demonstration of the chromosome X decoding process. Panel (c) provides an illustrative demonstration of the random-key solution decoding process for the TSP. Each solution is encoded as a vector of random keys of length $n = |N|$, where the i -th random key corresponds to the i -th city of N . The decoder starts by ordering the random-key vector in ascending order. Then, using this sequence, the cheapest insertion constructive heuristic is performed. A sub-route is created with the first three cities, and from the fourth city in the sequence, the cheapest position is calculated to insert the current city. The cost of inserting a city k after city j is calculated by $c_{k,j} = d_{j,k} + d_{k,j+1} - d_{j,j+1}$, where $d_{i,j}$ is the distance or costs between the cities i and j . In the end, the cost of the TSP route is computed.

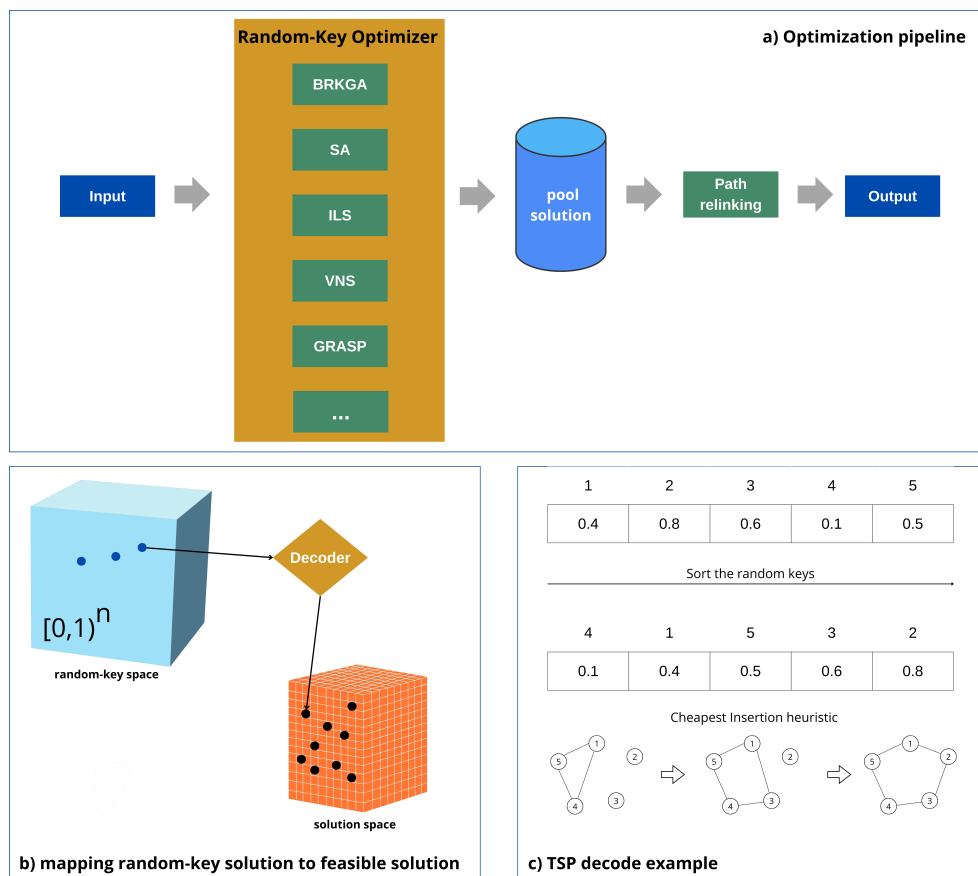


Figure 1: Schematic illustration of the RKO.

3. Objectives

This course aims to develop metaheuristics using the concept of random keys for combinatorial optimization problems, enabling participants to learn the basic concepts of some classic metaheuristics and how to adapt them for use within RKO.

The course will be practical in implementing these metaheuristics using the C++ programming language to solve an optimization problem.

4. Topics

1. Characteristics and classification of heuristic methods:
 - Constructive heuristics
 - Local search heuristics
 - Metaheuristics
2. Combinatorial optimization problems
3. Meta-heuristics: BRKGA, Simulated Annealing, GRASP, VNS, ILS, PSO, Path-Relinking
4. Random-key Optimizers (RKO)
5. Empirical analysis of heuristic methods

References

- Andrade, C. E., Toso, R. F., Gonçalves, J. F., and Resende, M. G. (2021). The multi-parent bi-ased random-key genetic algorithm with implicit path-relinking and its real-world applications. *European Journal of Operational Research*, 289(1):17–30.
- Bean, J. C. (1994). Genetic algorithms and random keys for sequencing and optimization. *ORSA Journal on Computing*, 6(2):154–160.
- Gonçalves, J. F. and Resende, M. G. (2011). Biased random-key genetic algorithms for combinato-rial optimization. *Journal of Heuristics*, 17(5):487–525.
- Oliveira, B. B., Carravilla, M. A., Oliveira, J. F., and Resende, M. G. C. (2022). A C++ application programming interface for co-evolutionary biased random-key genetic algorithms for solution and scenario generation. *Optimization Methods and Software*, 37(3):1065–1086.
- Schuetz, M., Brubaker, J., Montagu, H., van Dijk, Y., Klepsch, J., Ross, P., Luckow, A., Resende, M., and Katzgraber, H. (2022). Optimization of robot trajectory planning with nature-inspired and hybrid quantum algorithms. *Physical Review Applied*, 18(5).
- Toso, R. and Resende, M. (2015). A C++ application programming interface for biased random-key genetic algorithms. *Optimization Methods and Software*, 30(1):81–93.