Applied Soft Computing 46 (2016) 317–327

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Local search versus Path Relinking in metaheuristics: Redesigning Meta-RaPS with application to the multidimensional knapsack problem

Arif Arin[a], Ghaith Rabadi[b,*,1]

[a] Middle East Technical University, Department of Business Administration, Universiteler Mahallesi, Dumlupinar Bulvari No. 1, Ankara 06800, Turkey
[b] Engineering Management & Systems Engineering, Old Dominion University, Engineering Systems Building, Room 2102, Norfolk, VA 23529, USA

ARTICLE INFO

Article history:
Received 16 July 2015
Received in revised form 9 April 2016
Accepted 8 May 2016
Available online 18 May 2016

Keywords:
Local search
Machine learning
Path Relinking
Meta-RaPS
0–1 multidimensional knapsack problem

ABSTRACT

Most heuristics for discrete optimization problems consist of two phases: a greedy-based construction phase followed by an improvement (local search) phase. Although the best solutions are usually generated after the improvement phase, there is usually a high computational cost for employing a local search algorithm. This paper seeks another alternative to reduce the computational burden of a local search while keeping solution quality by embedding intelligence in metaheuristics. A modified version of Path Relinking is introduced to replace the local search in the improvement phase of Meta-RaPS (Meta-Heuristic for Randomized Priority Search) which is currently classified as a memoryless metaheuristic. The new algorithm is tested using the 0–1 multidimensional knapsack problem, and it is observed that it could solve even the largest benchmark problems in significantly less time while maintaining solution quality compared to other algorithms in the literature.

1. Introduction

As a more efficient alternative to exact mathematical methods, metaheuristic methods are quite promising approaches in solving optimization problems in terms of result quality, problem sizes they solve, and computational effort consumed. Due to computational efficiency concerns, the need to find near-optimal solutions in an acceptable amount of time becomes the main reason for using heuristic approaches in most real world applications. In general, metaheuristics can be observed as the repetition of the two main phases: generation of solutions followed by solutions improvement. In the first phase, solutions are produced based on the principles of the algorithm by gradually constructing or forming the whole solution at once. Most of the time, the initial solution is not expected to include the attributes of a high quality solution, thus in the second phase, the algorithm requires improving the initial solution by implementing various types of local search techniques.

The local search starts the search from an initial point that may be constructed by another heuristic, an entirely random process, or a combination of a set of several initial points [77]. After leaving the initial point or points, the algorithm keeps moving to better neighbors while employing different local search approaches. Guided local search [103] dynamically changes its augmented objective function when optimized by a local search according to the local optima found; while Stochastic local search [51] uses randomization to ensure that the search process does not stagnate with unsatisfactory candidate solutions methods. Path-Relinking is a major enhancement to trajectory-based stochastic local search algorithms that generate a sequence of locally optimal solutions [92]. Iterated local search [97,68] generates the starting solution for the next iteration by perturbing the local optimum found at the current iteration instead of repeatedly applying the local search to randomly generated starting solutions. There are other approaches to a local search in the literature, such as Memetic Algorithms [81] referring to the combination of evolutionary algorithms with a local search; a Variable Neighborhood Search whose main feature is exploring the search space through the systematic exchange of neighborhood structures randomly [94]; a Memory-based Local Search Heuristic [71] to achieve a suitable tradeoff between intensification and diversification; a Reactive Local Search [50] that applies a memory found in the search to avoid repetition of search results; and parallel local search algorithms [23] to speed up the computations needed by engaging several processors and dividing the total amount of work among them. Despite impressive advances

* Corresponding author. Fax: +1 757 683 5640.
E-mail addresses: arif.arin@metu.edu.tr (A. Arin), grabadi@odu.edu (G. Rabadi).
1 www.ghaithrabadi.com.

http://dx.doi.org/10.1016/j.asoc.2016.05.016
1568-4946/

in systematic, complete search algorithms, local search methods in many cases represent the only feasible way for solving these large and complex instances. Hoos and Tsang [52] discuss various local search methods.

Although the best solutions of the algorithms are mostly generated in their improvement phase, there is usually a high computational cost due to the execution of local search algorithms. For many applications, local search techniques in the second phase consume more time than constructing solutions in the first phase. Some metaheuristics try to overcome this phenomenon by employing more intelligent strategies to construction and local search mechanisms. Intelligent optimization refers to a more extended area of research, including online and offline schemes based on the use of memory and learning, adaptation, and incremental development of models, experimental algorithmics applied to optimization, intelligent tuning, and reactive search in the design of metaheuristics. Reactive Search advocates the integration of machine learning techniques into search heuristics for solving complex optimization problems [12]. Reactive Search describes an immediate response to events during the search through an internal feedback loop for online adaptation. The algorithm keeps the ability to respond to different situations during the search process, but the adaptation is automated and executed while the algorithm runs on a single instance reflecting on its past experience.

In the literature, there are various metaheuristic approaches that involve different learning mechanisms in their structures [8]. The content of these mechanisms varies from one metaheuristic to another. While a tabu list represents memory in Tabu Search (TS), in most other metaheuristics such as Evolutionary Algorithms (EA) and Genetic Algorithms (GA), search memory is limited to a population of solutions. In Ant Colonies Optimization (ACO), the pheromone matrix is the main component of the search memory, whereas Estimation Distribution Algorithms (EDA) involve a probabilistic learning model that composes the search memory. The sophisticated version of TS includes longer term memory with associated intensification and diversification strategies. Glover and Laguna [43] define this approach as Adaptive Memory Programming (AMP) because it is based on exploiting the strategic memory components. Based on the AMP approach, Dréo et al. [27] present Adaptive Learning Search (ALS) in which the memorized data are not only the raw data input, but also the information on the distribution and, thus, on the solutions.

There are also successful hybrid applications where metaheuristics are empowered by intelligent approaches to improve their effectiveness, such as in TS with linear programming [31], GA [100], Simulated Annealing (SA) [112], and EA [110]; GA with adaptive local search scheme [115]; Evolutionary Programming (EP) with fuzzy systems [99] and with Reinforcement Learning [54]; ACO with fuzzy systems [113]; EDA with Neural Networks (NN) [117], and Variable Neighborhood Search [93]; and PSO with EDA [70], Memetic Algorithm [53], Artificial Bee Colony [65], and ACO and 3-Opt algorithms [73]. The motivation behind these hybridization applications is to exploit the complementary character of different optimization strategies and benefit from their synergy [17]. Such frameworks store and utilize various information related to search history in order to reach high quality solutions. These intelligent algorithms, however, typically require mechanisms that may likely increase the need for computational memory and time for the solution process in addition to the computation necessary for the local search. Arin and Rabadi [9] implemented a Path Relinking (PR) learning approach in which learning takes place only after producing solutions and does not require any memory matrix to be trained. They applied the basic form of PR with Meta-RaPS to solve the 0–1 multidimensional knapsack problem (MKP) and, although it obtained very good results compared to other algorithms, the local search in the improvement phase of Meta-RaPS was time consuming. PR generates a path between solutions linked by series of moves to incorporate attributes of the best solution that it learns while searching the solution space. In this paper, PR is designed more effectively to reduce the computational burden by replacing the local search phase in Meta-RaPS with a modified and enhanced version of PR.

The 0–1 MKP, a special case of the general linear 0–1 integer programming problem with nonnegative coefficients, is used in this paper as a test bed to evaluate the performance of the proposed algorithm. The 0–1 MKP is the generalized form of the classical knapsack problem (KP) in which there is a knapsack with an upper weight limit b, a set of n items with different profits c_j and weights a_j per item j. The problem is to select the items from the set such that the total profit of the selected items is maximized without exceeding b. If m knapsacks exist, the problem becomes the MKP in which each knapsack has a different upper weight limit b_i, and an item j has a different weight a_{ij} for each knapsack i. The objective is to find a set of items with maximal profit such that the capacity of each knapsack is not exceeded [36]. The MKP can be formulated as follows:

$$\text{Maximize} \quad \sum_{j=1}^{n} c_j x_j . \tag{1}$$

$$\text{Subject to} \quad \sum_{j=1}^{n} a_{ij} x_j \le b_i \quad i = 1, \ldots, m \tag{2}$$

$$x_j \in \{0, 1\}, \quad j = 1, \ldots, n \tag{3}$$

where x is a vector of binary variables such that $x_j = 1$ if item j is selected, and $x_j = 0$ otherwise. In the literature it is assumed that profits, weights, and capacities are positive integers. However they can be easily extended to the case of real values [75]. The MKP is NP-hard [38] and the number of constraints increases its difficulty. Although the classical KP is weakly NP-hard, the MKP is much more difficult even for $m = 2$. According to Wilbaut et al. [108], 0–1 MKP instances with 500 variables and 30 constraints cannot be solved optimally within a reasonable amount of computing time and memory requirement.

The MKP is often used as a platform to evaluate new metaheuristics. Algorithms proposed in the literature to solve MKPs can be grouped into two classes: exact and heuristic/metaheuristic algorithms [102]. Exact techniques include Lagrangian methods and surrogate relaxation techniques, special enumeration techniques and reduction schemes, and branch-and-bound. In terms of metaheuristics, GA, GRASP (Greedy Randomized Adaptive Search Procedure), NN, SA, and TS are the most common approaches. Battiti and Tecchiolli [13] solved the MKP instances by employing the Reactive TS with satisfactory performance. Moraga et al. [80] implemented Meta-RaPS and achieved good results when comparing their algorithm to both optimal solutions and other 0–1 MKP solution techniques such as SA, TS, GA, and 0-1 MKP heuristics. Dynamic programming based approach [11], exact methods [18] and heuristic methods [19,30] are among the recent approaches that have been applied to the 0–1 MKP. Wilbaut and Hanafi [107] proposed several convergent algorithms to solve a series of small sub-problems of 0–1 MKP generated by relaxations. Khemakhem et al. [60] combined TS with a dynamic and adaptive neighborhood search algorithm for the MKP that used a Linear Programming-based heuristic to generate a starting solution to a filter-and-fan (F&F) procedure which is an iterative local search method exploring the solution space by generating moves in a tree search fashion. Yoon and Kim [114] applied a memetic algorithm Lagrangian relaxation (MLH) and used the MKP to evaluate its performance. Kong et al. [61] developed a new binary coded version of Harmony Search (HS), a meta-heuristic that has been applied widely to continu-