Operations Research Perspectives 4 (2017) 123–135

Contents lists available at ScienceDirect

Operations Research Perspectives

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

Automatic instantiation of a Variable Neighborhood Descent from a Mixed Integer Programming model

Tommaso Adamo, Gianpaolo Ghiani, Emanuela Guerriero, Emanuele Manni*

Dipartimento di Ingegneria dell'Innovazione, Università del Salento, Via per Monteroni, Lecce 73100, Italy

ARTICLE INFO

Article history:
Received 18 November 2016
Revised 15 September 2017
Accepted 15 September 2017
Available online 19 September 2017

Keywords:
Mixed Integer Programming
Variable Neighborhood Descent
Semantic features

ABSTRACT

In this paper we describe the automatic instantiation of a Variable Neighborhood Descent procedure from a Mixed Integer Programming model. We extend a recent approach in which a single neighborhood structure is automatically designed from a Mixed Integer Programming model using a combination of automatic extraction of semantic features and automatic algorithm configuration. Computational results on four well-known combinatorial optimization problems show improvements over both a previous model-derived Variable Neighborhood Descent procedure and the approach with a single automatically-designed neighborhood structure.

1. Introduction

The design of a metaheuristic algorithm comprises a number of steps, including the definition of "good" neighborhood structures in the solution space as well as the "tuning" of a number of parameters characterizing the higher level search strategy. Typically, to make the metaheuristic efficient, both the local and the global improvement mechanisms must be tailored to: (a) the specific problem; (b) the distribution of the instances to be solved (i.e., the *reference instance population*). Other relevant factors are: the time limit for the exploration of a single neighborhood structure, the time limit for the whole procedure as well as the hardware at disposal. The design process may take days, weeks or even months and is typically done by human experts through a number of steps, including:

(a) problem analysis: the problem structure as well as the characteristics of the reference instance population are thoroughly examined in order to extract meaningful properties and features;
(b) literature scouting: the literature related to the same or similar problems is analyzed trying to identify algorithms and approaches that have proven to be successful;
(c) neighborhood structures design: a number of tentative neighborhood structures are defined. Typically, these tentative designs are characterized by a set of parameters and some sort of experimental design [1] is applied to determine the *best* parameters' values;
(d) experimentation: the tentative neighborhood structures are assessed on a sample (*training set*) extracted from the reference instance population. The results obtained in this phase may suggest some modifications of the tentative neighborhood structures.

The aim of our research is to develop, without any human intervention, "good" metaheuristics from a given Mixed Integer Programming (MIP) model. The ultimate goal is to generate automatically metaheuristics that may provide *human competitive* results [2]. Recently [3] introduced a three-step procedure, which automatically designs a single neighborhood structure from a MIP model: (1) a set of semantic features are automatically extracted from both the MIP model and a given feasible solution; (2) neighborhood design mechanisms are derived from the extracted features; (3) a "proper mix" of such mechanisms are searched during an automatic configuration phase. In this paper we take a different perspective and generalize the previous work in the context of an entire metaheuristic algorithm. More specifically, we focus on *Variable Neighborhood Descent* (VND) [4] and define a procedure that - based on a MIP model and a current feasible solution - determines automatically the size and the "shape" of the whole VND hierarchy of neighborhoods, as well as the time limits for their exploration through a general-purpose black-box MIP solver.

The remainder of the paper is organized as follows. Section 2 is devoted to a review of the literature relevant to our work. In

* Corresponding author.
E-mail addresses: tommaso.adamo@unisalento.it (T. Adamo), gianpaolo.ghiani@unisalento.it (G. Ghiani), emanuela.guerriero@unisalento.it (E. Guerriero), emanuele.manni@unisalento.it (E. Manni).

https://doi.org/10.1016/j.orp.2017.09.001

Section 3 we present the basic idea underlying the proposed approach and describe its main procedures. In Section 4 we discuss the computational results obtained on four well-known combinatorial optimization problems. In particular, we discuss the improvements provided by our automatically-designed VND with respect to both a previous model-derived Variable Neighborhood Descent procedure and the approach with a single automatically-designed neighborhood structure. Finally, conclusions follow in Section 5.

2. Literature review

Our work is related to several areas. First of all, it is related to the field of model-derived neighborhoods. Among these contributions, very relevant are those based on the *Local Branching* concept [5] in which spherical neighborhoods defined by appropriate non valid inequalities are explored by using an off-the-shelf MIP solver. In particular, for purely binary MIPs a neighborhood of the current solution includes all the solutions in which the number of variables changing value (i.e., the *Hamming distance*) does not exceed a given threshold. Danna et al. [6] introduced the *Relaxation-Induced Neighborhood*, which is defined by fixing the variables with the same values in both the incumbent and the optimal solution of the continuous relaxation. Then, after setting a cutoff equal to the objective value of the current incumbent, the neighborhood is explored by solving the sub-MIP on the remaining variables. Parisini and Milano [7] presented a search strategy called *Sliced Neighborhood Search* that considers randomly selected slices of spherical neighborhoods. Particularly relevant to our work are the contributions by Ghiani et al. [8] and Adamo et al. [3], that showed how to take advantage of a MIP compact formulation to automatically design efficient neighborhood structures, without any human analysis. In particular, Ghiani et al. [8] used unsupervised learning to automatically select “good” portions of the search space “around” a given feasible solution. Adamo et al. [3] proposed a procedure extracting some semantic features from a given MIP model. Based on the selected features, some neighborhood design mechanisms are automatically derived and, finally, a “proper mix” of such mechanisms are determined by running an automatic configuration algorithm on a training set representative of the reference instance population. This approach was recently extended by Adamo et al. [9], that allowed the Automatic Neighborhood Design algorithm to deal with an *ensemble* of Constraint Programming (CP) and MIP models.

Another area to which our paper is strongly related is that combining model-derived neighborhoods and heuristic search. Such approaches are typically referred to as *matheuristics* [10], which are defined as heuristic algorithms obtained by integrating metaheuristics and mathematical programming techniques. In particular, Hansen et al. [11] combined Local Branching with *Variable Neighborhood Search*, whereas Lazić et al. [12] proposed to solve 0–1 MIPs by a hybrid heuristic based on the principle of *Variable Neighborhood Decomposition Search*. Then, [13] devised a general matheuristic that decomposes the problem being solved in a master and a subproblem. The main characteristic of the approach is that it exploits features of the incumbent solution to generate one or more columns in the master problem. More recently, [14] proposed a general approach for combinatorial optimization termed *Construct, Merge, Solve & Adapt*, in which sub-instances of the original problem are first generated by repeatedly constructing probabilistic solutions and then solved by using a general-purpose MIP solver. In the context of CP, Van Hentenryck and Michel [15] showed how constraint-based local search algorithms can be synthesized from high-level models, at least for some application classes. Such synthesis is driven by the model structure, as well as the role and the semantics of each single constraint. In particular, the high-level model is first classified and then a solution algorithm is selected from a predefined portfolio. These ideas have been exploited by Elsayed and Michel [16] to develop a model-driven automatic search procedure generator written in Comet [17]. The generator examines a CP model instance, analyzes the constraints as well as the variable declarations and synthesizes a procedure that is likely to yield good performances on the considered instance. More recently, Mouthuy et al. [18] showed how these concepts can be applied to a *Very Large Scale Neighborhood Search* framework, whereas Kiziltan et al. [19] combined constraint propagation with the Local Branching general-purpose neighborhood.

An alternative approach involves the use of hyper-heuristics [20] in which, given a particular problem instance, an appropriate low-level heuristic is selected from a given set and applied at each step. For instance, in the context of genetic programming [21] developed a system that uses a simple composition operator to automatically discover local search heuristics for the boolean satisfiability testing problem. A recent research trend [22,23] is to employ human-designed heuristics as building-blocks to automatically generate new heuristics suited to a given problem or class of problems.

Finally, our work is also related to *Automatic Algorithm Configuration* (AAC) [24–26], in which an automatic procedure finds the parameter configurations for which the empirical performance on a given set of problem instances is optimized. Nowadays, many AAC software packages have been developed, such as F-Race [27,28], Calibra [29], ParamILS [30] and irace [31]. AAC has also been used in combination with grammar representations, as in [32], where the authors proposed a novel representation of the grammar by a sequence of categorical, integer, and real-valued parameters. Then, they used an AAC tool to search for the best algorithm for the problem at hand.

3. Automatic instantiation of a variable neighborhood descent

As stated before, the aim of our research is to develop automatically “good” metaheuristics from a MIP model. The basic idea is to try to reproduce the behavior of a human researcher that must develop a neighborhood search heuristic for a given combinatorial optimization problem $\mathcal{P}$. The starting point is a knowledge base of the problem, that is thoroughly examined in order to identify the main components that are combined to obtain a feasible solution. For instance, in a vehicle routing problem such components would be the customers and the vehicles that must be associated each other to define the routes. Of course, the solution is feasible if customers to vehicles assignment satisfies some requirements (constraints) that are identified from the problem description. After defining the structure of a feasible solution, the researcher must identify neighborhood relationships on the search space that allow to move from a current solution to a new (possibly improving) one. For instance, in a vehicle routing problem a new solution could be obtained by moving a given number of customers from a route to another.

Analogously to a human-like approach, we must define a way to represent the knowledge associated to the problem that allows to easily and automatically identify and extract the main components of the problem as well as the relationships among them (*semantic features*, in the following). In this paper, we assume that $\mathcal{P}$ is given a *factored* representation (in which each state is coded by a fixed set of variables, see [33]) as a mixed-integer program. Let $J = \mathcal{C} \cup \mathcal{G} \cup \mathcal{B} = \{1, \dots, n\}$ be the variable index set ($\mathcal{C}$, $\mathcal{G}$, and $\mathcal{B}$ denote the index sets for continuous, general integer and binary variables, respectively). A generic MIP model for $\mathcal{P}$ can be stated