Computers & Operations Research 52 (2014) 170–180

Contents lists available at ScienceDirect

Computers & Operations Research

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

Scheduled penalty Variable Neighborhood Search

Barrett W. Thomas [a,*], Emanuele Manni [b]

[a] Department of Management Sciences, University of Iowa, Iowa City, IA, USA
[b] Dipartimento di Ingegneria dell'Innovazione, Università del Salento, Lecce, Italy

ARTICLE INFO

Available online 16 December 2013

Keywords:
Variable Neighborhood Search
Scheduled penalty
Traveling salesman problem with time windows
Orienteering Problem with Time Windows

ABSTRACT

For many $\mathcal{NP}-$hard combinatorial optimization problems, the existence of constraints complicates the implementation of a heuristic search procedure. In this paper, we propose a general heuristic framework that extends the well known Variable Neighborhood Search algorithm to include dynamic constraint penalization. We specifically focus on what are known as scheduled penalty methods and call the new algorithm scheduled-penalty Variable Neighborhood Search. The proposed method is tested on two well known constrained combinatorial optimization problems, namely the Traveling Salesman Problem with Time Windows and the Orienteering Problem with Time Windows. Our results demonstrate the effectiveness of the proposed algorithm, which is capable of producing high-quality solutions to the well known benchmark problems chosen in this paper with only minimal problem-specific tailoring of the general algorithm. Moreover, we introduce new best known solutions for some instances from the Orienteering Problem with Time Windows literature.

1. Introduction

For many $\mathcal{NP}-$hard combinatorial optimization problems, the existence of constraints complicates the implementation of a heuristic search procedure. Standard approaches for handling these constraints include (i) preserving feasibility in the search sequence of solutions through appropriately designed move operators and/or problem encodings, (ii) repairing infeasible solutions to obtain a corresponding feasible solution, and (iii) penalizing infeasibility to direct the search towards feasible solutions. In many instances, preserving feasibility may be difficult and repairing infeasibility may be inefficient, motivating the use of penalty functions to relax constraints. The notion of a natural neighborhood for local search algorithms can be regained through such constraint relaxation.

In general, penalty methods require the determination of the appropriate value for the penalty multiplier associated with each penalty term. The simplest approach is to consider a static penalty multiplier for each penalty term. For heuristics with static penalty multipliers, either elaborate procedures are required to determine appropriate multiplier values, or the penalties must be calibrated via extensive experimentation. As Costa and Oliveira [1] note, this is a major drawback as it can be difficult to determine the correct weighting factors for the different penalty terms. To overcome this difficulty, a number of authors have explored dynamic penalty methods that manipulate penalty multipliers within heuristic search methods.

In this paper, we extend the well known Variable Neighborhood Search algorithm (VNS) to include dynamic constraint penalization. We specifically focus on what are known as scheduled penalty methods and call the new algorithm scheduled-penalty VNS (spVNS). A scheduled penalty increases the level of penalization at each iteration of the search according to a predetermined schedule. The iterative increase of the penalty has the effect of gradually driving the search from infeasible to feasible regions of the search space.

This paper makes the following contributions to the literature:

- the introduction of a scheduled penalty VNS algorithm that blends important algorithmic features of VNS with features from other scheduled-penalty heuristics,
- a demonstration that the proposed algorithmic choices are capable of producing high-quality solutions to well known benchmark problems with only minimal problem-specific tailoring of the general algorithm, and
- as a minor contribution, an introduction of new best known solutions for some instances from the Orienteering Problem with Time Windows literature.

We outline the remainder of the paper as follows. In Section 2, we review the literature related to VNS, penalty methods, and the joint use of penalty methods and VNS. In Section 3, we describe our solution approach, and in Section 4, we present the test

* Corresponding author.
E-mail addresses: barrett-thomas@uiowa.edu (B.W. Thomas), emanuele.manni@unisalento.it (E. Manni).

0305-0548/$ - see front matter © 2014 Published by Elsevier Ltd.
http://dx.doi.org/10.1016/j.cor.2013.12.004

problems that we use to test the effectiveness of the proposed algorithm. Computational results follow in Section 5. Finally, in Section 6, we report our conclusions as well as some insights on future research directions.

2. Literature review

In this section, we present a brief description of VNS, as well as a review of the main contributions related to the use of penalty-based methods with a particular focus on those that have been used in conjunction with VNS.

2.1. Variable Neighborhood Search

Variable Neighborhood Search (VNS) was formally introduced in Mladenović and Hansen [2]. The basic idea is to explore increasingly distant neighborhoods of the current incumbent solution, both in a descent phase and a perturbation phase. This metaheuristic was originally designed for solving combinatorial optimization problems, and then extended to tackle mixed integer programs, nonlinear programs, and mixed integer nonlinear programs. Given a set $\mathcal{N}_k$ $(k = 1, \ldots, k_{\max})$ of pre-determined neighborhood structures, and denoted by $\mathcal{N}_k(x)$ the set of solutions in the kth neighborhood of a solution x, VNS is based on the observation that: (a) a local minimum with respect to one neighborhood structure is not necessarily so for another; (b) a global minimum is a local minimum with respect to all possible neighborhood structures; (c) for many problems, local minima with respect to one or several $\mathcal{N}_k$ are relatively close to each other. For a comprehensive review of VNS, the reader is referred to recent surveys by Hansen et al. [3,4].

2.2. Penalty methods

Penalty methods are a well known technique to handle constrained optimization problems. Penalty methods transform a constrained problem into an unconstrained one by adding a penalty term to the original objective function. The advantage of penalty methods is their simplicity and simple implementation.

Penalty methods in heuristic optimization can be characterized as either static or dynamic. A static penalty is a penalty that is fixed at the start of the search and never updated. A dynamic penalty is updated throughout the search.

While static penalties have been employed successfully (see [5–8]), Ohlmann and Thomas [9] demonstrate that static penalties often require problem-specific tuning and often fail to even find feasible solutions. Dynamic penalty methods have been shown to overcome this difficulty. We can typically classify dynamic penalties as either scheduled or adaptive. A scheduled penalty is one that evolves in a prespecified manner. An adaptive penalty is a penalty that changes in response to feedback from the search environment.

Scheduled penalty methods are most often associated with simulated annealing. Theodoracatos and Grimsley [10] extend simulated annealing with an ad hoc introduction of a variable penalty multiplier to complement the traditional simulated annealing parameter called temperature. These annealing-like penalty functions begin by minimally penalizing infeasible solutions and gradually increasing the penalty term over the course of the search. Ohlmann et al. [11] formalize this approach, referring to the value of the penalty multiplier as “pressure” and alluding to the resulting penalty-based annealing algorithm as “compressed” annealing. Ohlmann et al. [11] prove the conditions for convergence of compressed annealing when relaxing a set of constraints with a single penalty multiplier. Ohlmann and Thomas [9] demonstrate the effectiveness of compressed annealing on the TSPTW, and López-Ibáñez et al. [12] demonstrate the effectiveness of compressed annealing on the TSPTW with a makespan objective.

In addition to annealing algorithms, Carlson [13], Joines and Houck [14], Michalewicz and Attia [15] and Petridis and Kazarlis [16] combine annealing-like scheduled penalties with evolutionary algorithms. Mendivil and Shonkwiler [17] identify conditions under which genetic algorithms with annealing-like penalties will converge in probability. We note that each of the above works focuses on a single penalty parameter for all constraints.

Adaptive penalties have been used extensively in both evolutionary computation and tabu search. Early work includes Coit et al. [18], Ezziane [19], Hadj-Alouane and Bean [20], and Wu and Lin [21]. More recently, Tessema and Yen [22] propose a self-adaptive constrained optimization algorithm that is free of any parameter tuning. The main objective of the algorithm is to efficiently exploit the information hidden in infeasible individuals by selecting the proper individuals at different stages of the evolutionary process and under different conditions. Paszkowicz [23] seeks to computationally identify properties of genetic algorithms and adaptive penalty functions. Barbosa and Lemonge [24] propose an adaptive penalty method for genetic algorithms, in which an adaptive scheme automatically sizes the penalty parameter corresponding to each constraint along the evolutionary process. Ali et al. [25] provide a computational exploration of static and adaptive penalty methods for the population-based electromagnetism-like method and conclude that adaptive penalty methods offer the advantages described previously. Exhaustive surveys of penalty methods in evolutionary computation can be found in Coello Coello [26] and Yeniay [27].

A well known example of adaptive penalties in tabu search can be found in Gendreau et al. [28] who introduce an adaptive penalty tabu search for the vehicle routing problem. The method adjusts the penalty using a fixed scheme based on whether the current solution is feasible or not. Gendreau et al. [29] apply a similar adaptive penalty tabu search to a stochastic vehicle routing problem. Cordeau et al. [30] develop an adaptive penalty tabu search for the vehicle routing problem with time windows (VRPTW). In contrast to Gendreau et al. [28], Cordeau et al. [30] adjust the penalty size with regard to the frequency of its attributes and a scaling factor rather than simply the presence of infeasibility. Gopalakrishnan et al. [31] present an adaptive tabu search approach for a capacitated lot-sizing problem with set-up carryover. Kulturel-Konak et al. [32] introduce an adaptive penalty, based on Coit et al. [18], that penalizes solutions based on their distance from the feasible region and the search history.

We also note the use of adaptive penalty methods with other popular metaheuristic approaches. For example, Schlüuter et al. [33] use an adaptive penalty scheme known as an oracle method (see [34]) to solve non-linear mixed integer problems using ant-colony optimization. For a particle swarm heuristic, Masuda et al. [35] introduce an adaptive penalty that has similarities to the scheduled penalties discussed in Ohlmann et al. [11]. Özbakir et al. [36] introduce a modified Bees algorithm that uses an adaptive penalty to solve the generalized assignment problem.

2.3. Penalty methods and variable neighborhood search

In this section, we focus on papers that combine the use of penalty-based methods with a VNS heuristic. All of the methods employ either static or adaptive penalty methods. Burke et al. [37] and Hemmelmayr et al. [38] make use of a static penalty with a VNS. Works combining VNS and adaptive penalties are developed in Polacek et al. [39], Liang and Chen [40], Mladenović et al. [41], Hemmelmayr et al. [42], Trautsamwieser and Hirsch [43], and