Chapter 11
Designing Parallel Meta–Heuristic Methods

Teodor Gabriel Crainic
Département de Management et Technologie, Université du Québec à Montréal, CIRRELT, Canada

Tatjana Davidović
Mathematical Institute, Serbian Academy of Science and Arts, Serbia

Dušan Ramljak
Center for Data Analytics and Biomedical Informatics, Temple University, USA

ABSTRACT

Meta-heuristics represent powerful tools for addressing hard combinatorial optimization problems. However, real life instances usually cannot be treated efficiently in "reasonable" computing times. Moreover, a major issue in meta-heuristic design and calibration is to provide high performance solutions for a variety of problems. Parallel meta-heuristics aim to address both issues. The objective of this chapter is to present a state-of-the-art survey of the main parallelization ideas and strategies, and to discuss general design principles applicable to all meta-heuristic classes. To achieve this goal, the authors explain various paradigms related to parallel meta-heuristic development, where communications, synchronization, and control aspects are the most relevant. They also discuss implementation issues pointing out the characteristics of shared and distributed memory multiprocessors as target architectures. All these topics are illustrated by the examples from recent literature related to the parallelization of various meta-heuristic methods. Here, the authors focus on Variable Neighborhood Search and Bee Colony Optimization.

INTRODUCTION

Meta-heuristic methods are widely used for solving various combinatorial optimization problems. Computing optimal solutions is intractable for many important industrial and scientific optimization problems. Therefore, meta-heuristic algorithms are used for practical applications, since, in the majority of cases, they provide high quality solutions within reasonable CPU times. However, as the problem size increases, the execution time required by a meta-heuristic to find a high quality solution may become unreasonably long. Parallelization has proven to be an efficient method for overcoming this problem.

Since meta-heuristics usually represent stochastic search processes, it is important to properly define measures to evaluate the performance of their parallelized versions, as we cannot use the standard performance measures (speedup and efficiency) for the evaluation of parallel meta-heuristics. Actually, parallelization changes the original algorithm, and consequently, the evaluation of both *the execution time* and *the quality of the final solution* is needed. Indeed, for the majority of parallelization strategies, the sequential and parallel versions of heuristic methods yield solutions that differ in value, composition, or both. Thus, an important objective when parallel heuristics are considered is to design methods that outperform their sequential counterparts in terms of solution quality and, ideally, computational efficiency. More precisely, the parallel method should not require a higher overall computation effort than the sequential method, or should justify the effort by a higher quality of the final solutions. Consequently, we select the execution time and the quality of the final solution as our performance measures.

A significant amount of work has been performed in defining, implementing, and analyzing parallelization strategies for meta-heuristics. According to the survey papers (Verhoeven & Aarts, 1995; Cung, Martins, Ribeiro, & Roucairol, 2002; Crainic & Toulouse, 2010), several main ideas related to the parallelization of meta-heuristic methods can be recognized: starting from the low level parallelization realized by distributing neighborhoods among processors, up to the cooperative multi-thread parallel search (Crainic & Hail, 2005; Crainic & Toulouse, 2010). Many parallelization strategies dealing with various meta-heuristic methods have been proposed in recent literature (Verhoeven & Aarts, 1995; Toulouse, Crainic, & Gendreau, 1996; Ferreira & Morvan, 1997; Cung, Martins, Ribeiro, & Roucairol, 2002; Crainic & Hail, 2005; Crainic & Toulouse, 2010). The rich collection of papers on parallel meta-heuristics (Alba, 2005) is devoted to both theoretical and practical aspects of this topic. Here, we briefly recall important issues, and then focus on two main classes of meta-heuristics: neighborhood-based and population-based methods. We select a representative for each class and give a survey of the existing approaches to their parallelization. As the representative for neighborhood-based meta-heuristic methods we select Variable Neighborhood Search (hereinafter: VNS) because of its numerous successful sequential and parallel applications. Parallelization of population-based methods is illustrated on Bee Colony Optimization (hereinafter: BCO), the nature-inspired meta-heuristic recently on the rise.

The rest of this chapter is organized as follows. The next section contains a brief overview of meta-heuristic methods. Review of recent literature addressing the parallelization strategies for various meta-heuristic methods is presented in Section 3. The parallelization strategies proposed in recent literature for VNS are described in Section 4, while Section 5 contains the review of the parallel BCO method. Section 6 concludes the chapter.

META-HEURISTICS

Combinatorial optimization problems involve the selection of the best among (finitely many) feasible solutions. Each problem is defined by a set of objects, with associated contributions; an objective function computing the value of a particular subset or order of objects; and the feasibility rules specifying how subsets/orderings may be built. The best solution is the one satisfying feasibility rules and yielding the value of the objective function which is the highest/lowest among all possible combinations. This solution is called *the optimal solution* (*optimum*). For a more formal definition, let us assume that one desires to minimize an objective function $f(x)$, linear or not, subject to $x \in X \subseteq \mathbb{R}^n$. The set X collects constraints on the decision variables x and defines the feasible domain. Decision variables

Related Content

Virtualization and Cloud Computing: Business Models in the Virtual Cloud

Chaka Chaka (2015). *Cloud Technology: Concepts, Methodologies, Tools, and Applications (pp. 1687-1701).*

www.irma-international.org/chapter/virtualization-and-cloud-computing/119927

Fake Review Detection Using Machine Learning Techniques

Abhinandan V., Aishwarya C. A.and Arshiya Sultana (2020). *International Journal of Fog Computing (pp. 46-54).*

www.irma-international.org/article/fake-review-detection-using-machine-learning-techniques/266476

Performance Evaluation of Data Intensive Computing In the Cloud

Sanjay P. Ahujaand Bhagavathi Kaza (2015). *Cloud Technology: Concepts, Methodologies, Tools, and Applications (pp. 1901-1914).*

www.irma-international.org/chapter/performance-evaluation-of-data-intensive-computing-in-the-cloud/119939

A Survey on Research Initiatives for Healthcare Clouds

Rahul Ghosh, Ioannis Papapanagiotouand Keerthana Boloor (2014). *Cloud Computing Applications for Quality Health Care Delivery (pp. 1-18).*

www.irma-international.org/chapter/a-survey-on-research-initiatives-for-healthcare-clouds/110426

Fog Computing Quality of Experience: Review and Open Challenges

William Tichaona Vambe (2023). *International Journal of Fog Computing (pp. 1-16).*

www.irma-international.org/article/fog-computing-quality-of-experience/317110