Cooperative Co-Evolution and MapReduce:
A Review and New Insights for Large-Scale Optimisation

A. N. M. Bazlur Rashid, Edith Cowan University, Australia
Tonmoy Choudhury, Edith Cowan University, Australia

ABSTRACT

Real-word large-scale optimisation problems often result in local optima due to their large search space and complex objective function. Hence, traditional evolutionary algorithms (EAs) are not suitable for these problems. Distributed EA, such as a cooperative co-evolutionary algorithm (CCEA), can solve these problems efficiently. It can decompose a large-scale problem into smaller sub-problems and evolve them independently. Further, the CCEA population diversity avoids local optima. Besides, MapReduce, an open-source platform, provides a ready-to-use distributed, scalable, and fault-tolerant infrastructure to parallelise the developed algorithm using the map and reduce features. The CCEA can be distributed and executed in parallel using the MapReduce model to solve large-scale optimisations in less computing time. The effectiveness of CCEA, together with the MapReduce, has been proven in the literature for large-scale optimisations. This article presents the cooperative co-evolution, MapReduce model, and associated techniques suitable for large-scale optimisation problems.

KEYWORDS

Big Data, Computational Techniques, Distributed Evolutionary Algorithms, Divide-and-Conquer, Large-Scale, Meta-Heuristics, Optimisation Problems, Parallel Programming, Problem Decomposition

INTRODUCTION

There is a vast number of real-world optimisation problems classified as complex and large-scale because of their different characteristics, such as high-dimensions, massive volume, varieties of data, non-linear, or multi-modal (Shi et al., 2016). Examples of such optimisation problems include the domain of bio-medical (e.g., genomic and post-genomic studies), health sciences (Malik et al., 2018, Saliha, 2018, Haried et al., 2019, Strang and Sun, 2019), scheduling and planning problems (e.g., production and manufacturing systems), engineering design (e.g., lamination and casing optimisation for the electric motor), smart factories (e.g., cyber-physical systems), logistics and transportation (e.g., routing problems, or packing and cutting), smart grids, cities, and homes – energy-aware system (e.g., reduction of emissions from coal-fired power plants), and large-scale coal supply chain (Liu et al., 2019, Ahsan and Rahman, 2018). The availability of large-scale problems and their data offer the research community new opportunities to discover new insights. Hence, knowledge management, knowledge discovery, or decision-making from these large-scale data is essential using appropriate modelling and techniques (Wang and Meng, 2018, Farsäter and Olander, 2019, Hadad et al., 2013, Ahrari and Haghani, 2019).

DOI: 10.4018/IJITPM.2021010102

Evolutionary algorithms (EAs) are the fundamental choice to tackle these optimisation problems. Within the literature, there are two sorts of approaches to EA: *non-decomposition* and *decomposition*. The performance of EA deteriorates when solving large-scale optimisation problems due to their associated features; hence, non-decomposition methods are not suitable for these problems (Sun et al., 2019). The main challenge to the large-scale optimisation problems is either triggered by complex behaviour of the objective function with the number of decision variables, or by the requirement of unacceptable computational time, for example, by a simulation model (Chung and Paredes, 2015, Rasoolimanesh et al., 2018). Further, due to the exponentially increased search space of large-scale optimisation problems, this results in a drop in local optima and eventually, fitness evaluation becomes computationally expensive (Omidvar et al., 2010).

Standard and traditional EAs or meta-heuristic algorithms cannot generate relevant results in a reasonable time with a vast search space. Distributed evolutionary algorithms (DEAs) provide opportunities to solve problems in less computational time with the help of a meta-heuristic algorithm, cooperative co-evolutionary algorithm (CCEA). DEA addresses a high-dimensional problem with a divide-and-conquer approach. DEA can decompose the large-scale optimisation problems into smaller sub-problems through the distributed co-evolution. The distributed environment allows a DEA to maintain the population diversity for preventing local optima; it also supports multi-objective search (Gong et al., 2015). The CCEA decomposes a large-scale problem into smaller sub-problems, evolves each sub-problem separately, and collaborates individuals from different sub-populations to build a complete solution for the problem. Accordingly, the fitness of an individual is evaluated based on the subjective fitness landscape. Here the evaluating individual interacts with other individuals from the rest of sub-populations, and the collaboration performance is assigned as fitness to the individual (Ebrahimpour et al., 2018). In particular, when a problem is non-separable, decomposition techniques impact the performance of the overall process. To deal with non-separable problems in a distributed environment, co-evolution and multi-agents DEA models are most suitable (Han and Trimi, 2018, Shou et al., 2019). These models can, however, be interfaced with other DEA models, such as master-slave (Dubreuil et al., 2006), island (Pierreval and Paris, 2000), cellular (Alba and Dorronsoro, 2005), hierarchical (Folino et al., 2008), or pool (Roy et al., 2009). Among DEA models, island models have been proven to improve the global search ability of traditional EAs, where many sub-populations can be deployed on isolated islands for maintaining more than one best individual. Further, individuals on different islands can evolve in various ways and islands interact with each other only when individuals from one island migrate to another island during a fixed interval (Gong et al., 2015).

An extension of CCEA, the constructive cooperative co-evolutionary (C^3) algorithm, has been proposed by Glorieux et al. (2017a) to solve highly complex large-scale optimisation problems. They evaluated the performance of C^3 on large-scale global high-dimensional optimisation benchmark problems (Saxena et al., 2017). When C^3 is embedded with other search techniques, such as differential evolution (DE) (Nguyen et al., 2019), it outperforms CCEA and different optimisation algorithms. C^3 depends on the decomposition of a problem (separable or non-separable) and an initial right feasible solution.

Further, the MapReduce programming model on an open-source platform was initially implemented by Google research (Dean and Ghemawat, 2008). It is a parallel programming model and executes computations by communicating with the Hadoop Distributed File System (HDFS). MapReduce is capable of large-scale handling data in a distributed setting in parallel with the available resources using its map and reduce features. Moreover, MapReduce has the characteristics of fault-tolerance, data locality, scalability, programming ease, and flexibility (Hashem et al., 2016). Using the MapReduce programming model, a developed framework or algorithm can be parallelised in a distributed platform to reduce the computational time (Zhou and Gao, 2018, Muehlburger et al., 2019). Advantages and disadvantages of the algorithms mentioned above, namely, DEA, CCEA, C^3, HDFS, and MapReduce, are summarised in the below Table 1.