Applied Soft Computing 36 (2015) 185–203

Contents lists available at ScienceDirect

Applied Soft Computing

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

A great deluge and tabu search hybrid with two-stage memory support for quadratic assignment problem

Adnan Acan *, Ahmet Ünveren

Eastern Mediterranean University, Computer Engineering Department, Gazimagusa, Turkey

ARTICLE INFO

Article history:
Received 30 October 2014
Received in revised form 18 May 2015
Accepted 26 June 2015
Available online 29 July 2015

Keywords:
Combinatorial optimization
Great deluge algorithm
Memory-based search
Metaheuristics
Quadratic assignment problem
Tabu search

ABSTRACT

A two-stage memory architecture is maintained within the framework of great deluge algorithm for the solution of single-objective quadratic assignment problem. Search operators exploiting the accumulated experience in memory are also implemented to direct the search towards more promising regions of the solution space. The level-based acceptance criterion of the great deluge algorithm is applied for each best solution extracted in a particular iteration. The use of short- and long-term memory-based search supported by effective move operators resulted in a powerful combinatorial optimization algorithm. A successful variant of tabu search is employed as the local search method that is only applied over a few randomly selected memory elements when the second stage memory is updated. The success of the presented approach is illustrated using sets of well-known benchmark problems and evaluated in comparison to well-known combinatorial optimization algorithms. Experimental evaluations clearly demonstrate that the presented approach is a competitive and powerful alternative for solving quadratic assignment problems.

1. Introduction

Combinatorial optimization problems are representative models of many real life optimization tasks. Location of facilities, finding shortest routes, staff and task scheduling, packing and covering and segmentation of images are a few within a large set of combinatorial optimization problems for which development of dedicated solution methods is still an area of hot research. A common nature of many combinatorial optimization problems is their NP-complete type of complexity which directly indicates the computational difficulty of obtaining exact solutions for these problems. In fact, exact solutions are possible for (very) small-size problem instances and finding an approximate solution with close-to-optimum quality is the fundamental problem to be solved in most of the time. Since advanced computational resources enable us to deal with larger-size and more complex problems, development of more efficient solution methodologies, in terms of solution quality and convergence speed, became the ultimate objectives of almost all optimization tasks.

Considering the exploration of a solution space using a particular metaheuristic method employing a set of search operators, the two challenging problems to deal with are early stagnation at a locally optimal solution and efficient guidance towards globally optimal solutions. In this respect, searching around multiple promising solutions, particularly at initial iterations of an algorithm, should clearly be a major concern for the design of search algorithms. For this purpose, memory methods using an archive of a number of promising solutions, or reference points, are proposed and they are shown to be useful through experimental evaluations. Accordingly, a memory-based search procedure has two fundamental objectives; one is to intensify the search around the potentially promising solutions found so far while the second is diversifying the exploration within the solution space by using more than one reference template within the maintained memory. This way, early stagnation at a locally optimal solution can be avoided by providing alternative directions for search and more efficient guidance towards globally optimal solutions can be provided by appropriate memory-based search operators and memory update methods.

Great deluge algorithm (GDA), which is first proposed by Dueck [1], is a local search algorithm similar to simulated annealing (SA) with the exception that acceptance rule is deterministic and controlled by a threshold parameter called the *Level*. It uses two other algorithmic parameters, namely, the number of iterations and an estimation of the best fitness value of the objective function. In its

* Corresponding author. Tel.: +90 392 6301068; fax: +90 392 3650711.
E-mail addresses: adnan.acan@emu.edu.tr (A. Acan), ahmet.unveren@emu.edu.tr (A. Ünveren).

http://dx.doi.org/10.1016/j.asoc.2015.06.061
1568-4946/

search in a solution space, GDA accepts a new solution if its fitness is better than the best solution found so far or the fitness is less than a dynamically updated upper bound, which is the *Level*. Estimated best fitness value of the objective function is usually set less than or equal to the fitness of the best solution known so far. *Level* is initially set to the fitness of the initial solution and is lowered iteratively by an additive parameter β computed as a function the initial value of *Level* and the estimated best fitness value of the objective function.

After its initial introduction, improved versions of GDA are proposed in several of publications. These improvements can be considered in two categories: changes in basic algorithm parameters and search operators, and hybridizations with other metaheuristics. Since GDA has basically two parameters, namely the *Level* and the level decay rate β, algorithmic improvements based on different ways of changing these parameters are the main concern of studies in the first category. Burke et al. implemented a linear decreasing of decay rate *beta* such that the amount of decay is computed based on the maximum number of iterations. The authors called this strategy as degraded ceiling algorithm [2]. According to published results, this strategy produced most of the best fitness values on examination timetabling benchmark problems. Later, Burke and Bykov developed the flex-deluge algorithm (FGDA) in which the acceptance of uphill moves is controlled by a flexibility parameter [3]. For the solution of exam timetabling problems, FGDA was claimed to provide good results particularly for large-scale problems. The modified great deluge algorithm (MGDA), proposed by Ravi [4], introduced a new neighbourhood search and a new level decay mechanism that depends on the amount of improvement achieved when the current solution is modified to get the new one. Basically, level is decreased if fitness of the new solution is below the current level, whereas it is increased otherwise. This approach is applied for reliability optimization and optimal redundancy allocation problems and was observed to perform as well as ant colony optimization, while it was significantly better than simulated annealing algorithm. Silva and Obit developed the nonlinear great deluge algorithm (NLGDA) that uses a nonlinear decay rate for the level parameter [5]. An exponential expression including four parameters is used to determine the functional shape of decay rate. When used for the solution of course timetabling problems, NLGDA updated the best solutions of four benchmark instances among the eleven in the experimental set. The extended great deluge algorithm (EGDA), developed by McMullan, uses the concept of reheating from simulated annealing such that, if no improvement is obtained within a predefined period, the level parameter is reset to the current objective function value [6]. For a set of course timetabling benchmark problems, EGDA produced ten best solutions for five medium-size problems and for one large-size problem. Nahas et al. used EGDA in two steps as follows: EGDA is used for N_1 number of iterations to find a locally optimal solution. Then, best found solution is used as the initial solution for EGDA and improved through another $N_2, N_2 < N_1$, number of iterations [7]. The authors called this approach as iterated great deluge algorithm (IGDA) and, for a set of 48 facility layout problems, IGDA is reported to update the best found results for 17 of them. Ozcan et al. combined reinforcement learning (RL) and the GDA within a hyperheuristic (HH) framework such that RL is used to select one of a set of low-level heuristics, that is applied as a move to generate a new solution, whereas the GDA's level-based decision mechanism is used to accept or reject the move [8]. For a set of thirteen university examination timetabling problems, this approach is observed to be better than simulated annealing and simple random hyper-heuristics for six problem instances.

The second category of the improvements includes hybridizations with other metaheuristics or slight modifications of level-based acceptance mechanism. Milli [9] used GDA as a local search procedure within a genetic algorithm (GA) framework for the solution of course timetabling problems. After each generation, GDA is employed to improve the best solution found so far. The author claimed that the proposed combination generated consistently good results for benchmark problems used in experimental studies. Landa-Silva et al. proposed another hybrid of evolutionary algorithms and NLGDA where an individual selected from the current population by tournament selection is modified by mutation and the mutated individual is improved by NLGDA [10]. If the resulting solution is better than the worst solution in GA's population, it replaces that worst solution. The proposed hybrid approach is claimed to perform better than its competitors in most of the benchmark course timetabling problems used for comparative evaluations. A third GDA-related hybrid approach on the solution of course timetabling problem combines particle collision algorithm (PCA) and GDA such that GDA's level-based acceptance criterion is used in the scattering phase of PCA [11]. The experimental results presented in the paper show that the proposed hybrid approach performs better than its eleven competitors for the solution of eleven widely used course timetabling problem instances. In a sequential implementation of Tabu search and GDA, proposed by Abdullah et al. [12], the current solution is first modified by GDA and then the modified solution is improved by tabu search. Finally, the best of the solutions found by the two algorithms is used to start the next iteration. When used for the solution of course time tabling problems, the proposed method is reported to provide the best solutions for most of the benchmark problems. In another publication, Abdullah et al. combined GDA with an electromagnetic-like mechanism to solve the examination timetabling problem [13]. This population based method is implemented in two phases. In the first phase, a positive amount of charge is assigned to each timetable in the population based on their fitness relative to the fitness of the best timetable found so far. Consequently, the total force on each timetable is computed using an analogy to interaction of charged particles and these total force values are used to calculate the estimated qualities of individual timetables in the population. In the second phase, the level decay parameter of GDA is determined based on the estimated qualities of individual timetables and GDA is applied on each timetable of the population for a number of iterations. Among the eleven examination timetabling instances and against the eight competitors in experimental evaluations, the authors reported that this hybrid approach provided the best known solutions for nine problem instances.

Among a few GDA hybrids used for real-valued function optimization, Ghatei et al. hybridized GDA and particle swarm optimization (PSO) in which GDA is used as final local search procedure to further improve the best solution found by the PSO algorithm [14]. This approach is tested over four benchmark functions only and it is claimed to be better than GAs and PSO.

Considering the above literature review and based on our detailed search for different solution approaches to QAP, a direct application of GDA for the solution of QAP could not be found so far. However, a problem that is most similar to QAP and solved using GDA is the dynamic facility layout problem [7]. In addition to this, even though GDA is not used for the solution of QAP, its success is demonstrated on another subclass of large-size combinatorial optimization problems, namely the timetabling and scheduling problems. Hence, our main reasoning behind the use of GDA for the solution of QAP can be stated as its exhibited success for difficult classes of combinatorial optimization problems, implementation simplicity, adaptive and deterministic decision character, and the objective of demonstrating the effectiveness of a novel two-stage memory architecture within the framework of a metaheuristic for the solution of a provably difficult combinatorial optimization problem. The proposed two-stage memory-based and operator-adapted variation of GDA for QAP, its objectives and comparative level of success are illustrated in the following sections.