Applied Mathematical Modelling 39 (2015) 2703–2720

Contents lists available at ScienceDirect

Applied Mathematical Modelling

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

Grouping evolution strategies: An effective approach for grouping problems

Ali Husseinzadeh Kashan [a], Ali Akbar Akbari [b,*], Bakhtiar Ostadi [a]

[a] Department of Industrial Engineering, Faculty of Engineering, Tarbiat Modares University, Tehran, Iran
[b] Department of Industrial Engineering, South Tehran Branch, Islamic Azad University, Tehran, Iran

ARTICLE INFO

Article history:
Received 9 April 2012
Received in revised form 12 August 2014
Accepted 3 November 2014
Available online 22 November 2014

Keywords:
Grouping problems
Grouping genetic algorithm
Grouping evolution strategies
Grouping particle swarm optimization
Batch-processing machine scheduling problem
Bin-packing problem

ABSTRACT

Many combinatorial optimization problems include a grouping (or assignment) phase wherein a set of items are partitioned into disjoint groups or sets. Introduced in 1994, the grouping genetic algorithm (*GGA*) is the most established heuristic for grouping problems which exploits the structural information along with the grouping nature of these problems to steer the search process. The aim of this paper is to evaluate the grouping version of the classic evolution strategies (*ES*) which originally maintain the well-known Gaussian mutation, recombination and selection operators for optimizing non-linear real-valued functions. Introducing the grouping evolution strategies (*GES*) to optimize the grouping problems that are intrinsically discrete, requests for developing a new mutation operator which works with groups of items rather than scalars and is respondent to the structure of grouping problems. As a source of variation, *GES* employs a mutation operator which shares a same rationale with the original *ES* mutation in the way that it works in continuous space while the consequences are used in discrete search space. A two phase heuristic procedure is developed to generate a complete feasible solution from the output of the mutation process. An extensive comparative study is conducted to evaluate the performance of *GES* versus *GGA* and *GPSO* (a recently proposed grouping particle swarm optimization algorithm) on test problem instances of the single batch-processing machine scheduling problem and the bin-packing problem. While these problems share exactly a same grouping structure and the performance of *GES* on both problems is reliable, switching from one problem to another deteriorates the performance of *GGA*. Though such a deficiency is not observed in the performance of *GPSO*, it is still inferior to *GES* on the single batch-processing machine scheduling test problem instances. Beside such empirical outcomes, the paper conveys a number of core strengths that the design of *GES* supports them but the design of *GGA* does not address them.

1. Introduction

Falkenauer [1] defines a grouping or assignment type problem as the one where the aim is partitioning a set V of n items into a collection of mutually disjoint subsets (groups) G_i such that: $V = \bigcup_{i=1}^{D} G_i$ and $G_i \cap G_j = \emptyset,\ i \neq j$. The above definition says

* Corresponding author. Tel.: +98 9122478805; fax: +98 2177500506.
E-mail addresses: a.kashan@modares.ac.ir (A.H. Kashan), a_akbari@azad.ac.ir (A.A. Akbari), bostadi@modares.ac.ir (B. Ostadi).

http://dx.doi.org/10.1016/j.apm.2014.11.001
0307-904X/

that in a grouping problem the aim is to partition the members of V into D ($1 \leqslant D \leqslant n$) different groups such that each item being assigned exactly to one group. Originally it is assumed that the ordering of groups is not relevant in grouping problems. However, there are many grouping type problems in which the ordering of groups is important.

Many NP-hard combinatorial optimization problems such as graph coloring problem, bin-packing problem, batch-processing machine scheduling problem, line-balancing problem, timetabling problem, identical/non-identical parallel-machines scheduling problem, cell formation problem, pickup and delivery problem etc, are well used examples of grouping problems.

In most of grouping problems, not all possible groupings are permitted since the group formations must be in such a way that a number of constraints being satisfied. Besides grouping constraints, the groups are usually formed based on an objective function which is founded on the composition of groups. Therefore, the building blocks that should be taken into consideration in an evolutionary search should be the groups or the group segments, but not the items isolatedly.

In terms of the number of groups (D) in a given solution of a grouping problem, two categories of problems are recognizable. The *constant grouping problems* are those problems in which the number of groups (D) is an input constant to the problem. Here, in terms of the number of groups, all solutions are at the same length. An example of this type of grouping problems is the identical/non-identical parallel-machines scheduling problem in which a number of jobs should be processed just by one of D available machines working in parallel to minimize the makespan. Here the task is to decompose the set of available jobs into D subsets where each subset i contains all task that should be processed by machine i. As can be seen any solution to this problem must partition the set of jobs into D subsets. On the other side, there are grouping problems in which, D is not known in advance and the objective is to find a feasible grouping yielding the minimum D. Let us refer to these problems as *variable grouping problems.* Bin-packing problem, single batch-processing machine scheduling problem and graph coloring problem are some examples of variable grouping problems.

We can further classify the grouping problems based on the type of groups, e.g., *identical* or *non-identical.* A grouping problem is referred to as the one with *non-identical groups* when the groups differ in their characteristics. If we exchange the whole content of two groups in a given solution of such problems, the resultant grouping differs from the original grouping. For example, let us consider the non-identical parallel-machines scheduling problem in which the processor machines differ in their operational characteristics such as processing speed, cost, etc. Given the fact that the set of jobs assigned to each machine constitute a group, these groups are not identical in the sense that their corresponding possessors are different. A grouping problem with *identical groups* is the one in which all groups are similar in their characteristics. Here, the complete exchange of items between two groups does not change the situation and the ordering between groups may be irrelevant. The identical parallel-machines scheduling problem, bin-packing, single batch-processing machine scheduling and graph coloring problems are among the grouping problems with identical groups.

There are many grouping problems in which the ordering between groups is important and the quality of the solutions in terms of the problem objective is influenced by the way in which groups are ordered. Such grouping problems are referred to as the *order dependent* grouping problems [2]. An example of this family of problems is the university exam time tabling problem. In an intuitive way we can introduce the *order independent* grouping problems.

Falkenauer [1] explains that the most commonly used representations for grouping problems, e.g., number encoding and order-based representation suffer from redundancies. In the number encoding, the value of the kth gene represents the group that item k is in. For example the individual $\underline{21321}$ encodes the grouping in which the first item is in group 2, the second item in 1, the third item in 3, and so on. However, it is easy to check that the individual $\underline{12312}$ encodes exactly the same solution for a typical grouping problem with identical groups. Moreover, under number encoding, if some constraints on the groups exist, the resulting chromosomes of crossover stage will certainly contain many illegal groups. The indirect order-based representation uses a decoder to build solutions from permutations of the items. Drawbacks of such typical representations have been presented in [1,3]. To remedy these drawbacks, Falkenauer introduced the *group encoding* and used it in genetic algorithm (*GA*). The idea of group encoding is that the items belonging to the same group should be placed into the same partition. For instance, the above individual can be represented as {{2, 5}, {4, 1}, {3}}. Using this encoding scheme, the genetic operators can work on groups rather than items unlike in number encoding (note that the ordering within and between partitions is irrelevant). The rationale is that in grouping problems these are the groups that are the innate building blocks of the problem, which can convey information on the expected quality of the solution they are part of, and not the particular positions of any one item on its own. Therefore the representations and resulting evolutionary search operators need to be defined such that they allow the groupings of items to be propagated. For a review of suitable encoding representations for grouping problems, readers may refer to Ülker et al. [4,5]. With this in mind, a standard *grouping genetic algorithm* (*GGA*) has been proposed by Falkenauer in 1994 which is a genetic algorithm that uses group encoding and related operators for solving grouping problems. There has since been applications of *GGA* to a number of grouping problems, with varying degrees of success. Table 1 gives a list of problems where *GGA* has been successfully applied to them.

Almost all researches that have used group encoding and operators have only relied upon genetic algorithm (*GA*) as their evolutionary search mechanism. A lot of *GGA* methodologies have been emerged and adapted to different grouping problems, without any effort put on developing the grouping version of other meta-heuristics (e.g., simulated annealing (*SA*), tabu search (*TS*), evolution strategies (*ES*), particle swarm optimization (*PSO*) etc). The notion of group related encoding and operators can be simply applied to *SA* to obtain the grouping version of *SA* (*GSA*) (this can be done using *GGA* mutation