Expert Systems With Applications 116 (2019) 65–77

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Neuroevolution for solving multiobjective knapsack problems

Roman Denysiuk[a,*], António Gaspar-Cunha[a], Alexandre C.B. Delbem[b]

[a] IPC Institute for Polymers and Composites, University of Minho, Guimarães 4800-058, Portugal
[b] University of São Paulo, Institute of Mathematical and Computer Sciences, São Carlos, SP 13566-590, Brazil

ARTICLE INFO

Article history:
Received 6 February 2018
Revised 2 September 2018
Accepted 2 September 2018
Available online 3 September 2018

Keywords:
Evolutionary computation
Multiobjective knapsack problem
Neuroevolution

ABSTRACT

The multiobjective knapsack problem (MOKP) is an important combinatorial problem that arises in various applications, including resource allocation, computer science and finance. When tackling this problem by evolutionary multiobjective optimization algorithms (EMOAs), it has been demonstrated that traditional recombination operators acting on binary solution representations are susceptible to a loss of diversity and poor scalability. To address those issues, we propose to use artificial neural networks for generating solutions by performing a binary classification of items using the information about their profits and weights. As gradient-based learning cannot be used when target values are unknown, neuroevolution is adapted to adjust the neural network parameters. The main contribution of this study resides in developing a solution encoding and genotype-phenotype mapping for EMOAs to solve MOKPs. The proposal is implemented within a state-of-the-art EMOA and benchmarked against traditional variation operators based on binary crossovers. The obtained experimental results indicate a superior performance of the proposed approach. Furthermore, it is advantageous in terms of scalability and can be readily incorporated into different EMOAs.

1. Introduction

The knapsack problem is a well-known combinatorial optimization problem. It comprises a knapsack of certain capacity and a set of items characterized by weights and profits. The aim is to fill up the knapsack by a collection of items so that the total profit is maximized and the total weight does not exceed the given capacity. There are several variants of a knapsack problem. The most common variant is the 0-1 knapsack problem, where the number of copies of each item is either zero or one. The bounded knapsack problem allows multiple copies of each item to be selected. Though, the maximum number of copies of each item is limited. On the contrary, the unbounded knapsack problem removes the restriction on the number of copies of each item. Generalizations of knapsack problem include multiple knapsack and multiobjective knapsack problems. The former arises when several knapsacks of given capacities are available and the total profit of all knapsacks is maximized. The latter implies simultaneous maximization of profits of several knapsacks, where for each knapsack different profit and weight are associated with each item. Knapsack is an important problem because it frequently occurs in practical applications including cryptography (Diffie & Hellman, 1976), resource allocation in distributed computer systems (Gavish & H.Pirkul, 1985), cargo loading (Pisinger, 1995), project and investment selection (Kellerer, Pferschy, & Pisinger, 2004).

Due to its practical and theoretical importance, the knapsack problem has been intensively investigated. There have been developed different exact and approximate algorithms. Exact methods are based on dynamic programming (Gilmore & Gomory, 1966; Weingartner & Ness, 1967), branch-and-bound (Gavish & H.Pirkul, 1985; Shih, 1979), and statistical analysis (Fontanari, 1995). The knapsack problem is NP-hard (Martello & Toth, 1990). In practice, this means there is no algorithm that exactly solves the problem in polynomial time. As a result, exact algorithms can be efficiently applied only to instances of moderate sizes. This fact has motived the use of heuristic and metaheuristic methods to find approximate solutions. There are various approximate solution developments such as greedy (Dantzig, 1957; Spielberg, 1969) and local search strategies (Petersen, 1974), tabu search (Glover & Kochenberger, 1996), simulated annealing (Drexl, 1988), genetic algorithm (Martins, Fonseca, & Delbem, 2014; Sakawa & Kato, 2003), ant colony (Kong, Tian, & Kao, 2007), harmony search (Zoua, Gaoa, Lib, & Wua, 2011) and artificial fish swarm (Azad, Rocha, & Fernandes, 2014) algorithms. A particular advantage of many metaheuristics is the ability to efficiently perform global search, although there is no guarantee of finding a global solution. Besides optimization methods, machine learn-

* Corresponding author.
E-mail addresses: roman@dep.uminho.pt (R. Denysiuk), agc@dep.uminho.pt (A. Gaspar-Cunha), acbd@icmc.usp.br (A.C.B. Delbem).

https://doi.org/10.1016/j.eswa.2018.09.004
0957-4174/

ing tools can be also applied to solving search problems. In particular, neural networks were employed to tackle several types of combinatorial optimization problems (Smith, 1999). The application of neural networks to solve knapsack problems are based on using Boltzmann machines (Vaithyanathan, Ogmen, & Ignizio, 1994), mean-field (Ohlsson, Peterson, & Soderberg, 1993), Hopfield (Yamamoto, Ohta, Ueda, Ogihara, & Fukunaga, 1995) and chaotic (Zhou, Kuang, & Wang, 2008) networks. Such approaches aim at exploiting learning capabilities of neural networks. They use a gradient-based learning to minimize the network energy function defined to be equivalent to the objective function of the knapsack problem.

The present study considers the 0-1 multiobjective knapsack problem. This is an important variant of knapsack problem that naturally arises in various application domains due to a multicriteria character of most real-world problems. MOKP inherits challenges of its single objective counterpart and the complexity of the objective space defined by multiple knapsacks. This study aims to address difficulties experienced by evolutionary multiobjective optimization algorithms on MOKPs such as the scalability and the loss of diversity (Ishibuchi, Akedo, & Nojima, 2010a; 2015; Ishibuchi, Narukawa, Tsukamoto, & Nojima, 2008; Sato, Aguirre, & Tanaka, 2007; Y.-Y. Tan, 2013). The main focus is on the mechanism for generating solutions to MOKP. For this, the use of neural networks is suggested. This idea attempts to exploit the ability of neural networks to compute a nonlinear model for given data and to perform classification. In the context of MOKP, the profits and weights of items are used as input. This input is propagated through the network to compute a scalar value in the output layer that classifies the item as either selected or not. Contrary to a traditional classification task, there are no target values that can be used for training. Therefore, traditional gradient-based learning cannot be performed and neuroevolution is suggested for setting the network parameters. In addition to the values of connection weights and biases, neuroevolution can learn the topology of neural network, thereby reducing the user's labor.

The remainder of this paper is organized as follows. Section 2 reviews the research concerned with the application of EMOAs to solve MOKPs. Section 3 formally defines the problem to be investigated. Section 4 presents the concepts exploited in this current study. Section 5 describes the proposed approach. Section 6 reports and discusses the results of the experimental study. Section 7 concludes the study and outlines some possible future work.

2. Solving MOKPs by EMOAs

Following the seminal work of Zitzler and Thiele (1999), there has been a significant amount of research in the field of evolutionary multiobjective optimization involving 0-1 MOKPs. This section starts by giving a general outline of EMOA and then discusses advances in the design of EMOAs for solving MOKPs. The discussion is organized according to constraint handling, selection and variation procedures.

2.1. EMOA framework

For the purpose of our discussion, we outline a general framework of EMOAs in Algorithm 1.

The initialization procedure is applied to initialize a population of solutions. This can be done uniformly at random within the feasible space. Thereafter, the population undergoes an evolutionary process that consists of the sequential application of the mating selection, variation and environmental selection procedures. The mating selection procedure picks up parents from population for reproduction giving a higher probability for fitter individuals, according to multiple criteria. The variation procedure is used for producing offspring. This is performed by evolutionary operators that are applied to the chromosomes of parent individuals. The environmental selection procedure forms the population of the next generation from the multiset of the current and offspring populations relying on the concept of the survival of the fittest from natural evolution. Most existing EMOAs can be described by the above algorithm, with major differences lying in the design of its procedures, as highlighted in the following subsections.

Algorithm 1 EMOA

```
1: P ← Initialization()
2: repeat
3:     R ← MatingSelection(P)
4:     Q ← Variation(R)
5:     P ← EnvironmentalSelection(P ∪ Q)
6: until stopping criterion is met
```

2.2. Constraint handling

As the definition of MOKP involves a set of constraints, the feasibility of solutions must be ensured. This can be addressed by penalizing an individual's fitness depending on the extent to which it violates the constrains. Another popular method consists of repairing an infeasible individual so that a feasible one is generated and used for fitness assignment. A common repair procedure for MOKP consists of removing items until all the constraint conditions are satisfied. The order in which items are removed is determined by a profit/weight ratio calculated for each item. This implies items having lower values are removed first. In MOKP, there are several profit/weight ratios associated with a single item, which complicates the repair procedure.

There have been proposed different ways to derive a scalar profit/weight ratio for an item. Zitzler and Thiele (1999) suggested a greedy repair mechanism where the maximum value among different profit/weight ratio was assigned to each item. Jaszkiewicz (2002) proposed a weighted scalar repair mechanism where a weighted sum of different profit/weight for each knapsack was assigned to each item (a weight vector is randomly generated each time an infeasible solution is repaired). Ishibuchi and Kaige (2003) investigated the effects of the two above mechanisms and the third where items are removed in the increasing order of the profit/weight ratio with respect to a randomly chosen knapsack. The results of this study showed that the performance of MOEAs on MOKPs is significantly affected by the type of repair mechanism.

In general, the repair procedure for dealing with infeasible solutions can be characterized by Lamarckian and Baldwinian schemes. The former implies that after the evaluation a feasible solution generated by the repair procedure replaces an infeasible solution in the population. The latter only repairs an infeasible solution for evaluation and assigns the corresponding fitness to the infeasible solution that remains unchanged in the population. Ishibuchi, Kaige, and Narukawa (2005) performed the comparison between both schemes in the context of multiobjective knapsack problem. The results of the study showed some superiority of Baldwinian scheme, although the tests have been performed only on a problem with two knapsacks and the authors suggested a further investigation in order to generalize the results.

2.3. Selection

There are mating and environmental selection procedures. They act relying on the fitness values of different population members

and a mechanism for their assignment. The selection and fitness assignment play an important role in addressing a major issue in multiobjective optimization - the balance between the convergence and diversity of solutions. With this regard, there can be identified two major strategies. The first is based on decomposing the objective space into a number of single objective subproblems. The fitness for individuals is calculated based on the parameters defining the location of a particular subproblem in the objective space. MOEA/D (Li & Zhang, 2009; Zhang & Li, 2007) is a popular algorithm adopting this idea. The second imposes some quality measure on a particular set of solutions and seeks solutions improving that measure. This is commonly achieved by using performance indicators (Zitzler & Künzli, 2004) or ranking solutions on the basis of the Pareto dominance relation (Deb, Pratap, Agarwal, & Meyarivan, 2002; Zitzler & Thiele, 1998a). The latter approach is typically used in combination with a diversity preserving mechanism that usually relies on distances between solutions (Zitzler, Laumanns, & Thiele, 2001) or quality measures (Beume, Naujoks, & Emmerich, 2007). It is noteworthy that both strategies can be used in combination (Li, Deb, Zhang, & Kwong, 2015; Liu, Gu, & Zhang, 2014).

It has been observed that dominance-based algorithms often suffer from the loss of diversity when handling MOKPs. This can be attributed to the combinatorial nature of the problem and the type of recombination operator in use. Most studies use a binary encoding together with traditional recombination operators such as one-point, two-point and uniform crossovers. To improve the performance of MOEAs, several studies suggest restricting the mating selection so that similar parents are used for recombination (Ishibuchi et al., 2008; Sato et al., 2007). The similarity between individuals is, in general, measured in the decision and/or the objective space using Hamming and Euclidean distances, respectively.

As opposed to many EMOAs, which necessitate modifications in order to control the similarity of parents, multiobjective evolutionary algorithm based on decomposition (MOEA/D) has an intrinsic mechanism enabling mating restriction and selection of similar parents. This mechanism is based on the neighborhood structure defined using the distances between weight vectors. Due to this feature, MOEA/D often exhibits promising performance on MOKPs (Ishibuchi, Akedo, & Nojima, 2015; Ishibuchi, Sakane, Tsukamoto, & Nojima, 2009; Zhang & Li, 2007). However, a special care may be required with respect to setting control parameters (Ishibuchi et al., 2009) and the choice of scalarizing function (Denysiuk & Gaspar-Cunha, 2017; Ishibuchi, Sakane, Tsukamoto, & Nojima, 2010b), since inadequate settings can lead to poor results.

Another important issue for the performance relates to the combinatorial nature of the problem. Ishibuchi, Yamane, and Nojima (2012) showed that a discrete objectives with a coarse granularity can slow down the search process. Also, some scalarizing functions can work well when the dimensionality of the objective space is low and experience the loss of diversity in higher dimensions. Sato (2014) addressed this issue by proposing an inverted penalty-based boundary intersection method for scalarization. Nonetheless, Ishibuchi et al. (2015) further pointed out the need to keep the population diversity.

2.4. Variation

Variation is a stochastic operator that acts on the genotype of population members to produce new individuals. It is responsible for the exploration of the search space and plays a crucial role in the performance of EMOAs. Naturally, numerous studies have focused on the development of variation operators to improve the performance on MOKPs, which is also the main concern of the present study study.

The design of a variation operator depends on the type of solution representation. Although various schemes have been investigated (Mumford, 2003), a binary solution representation is most frequently used for handling MOKPs, in part because it is easy to implement and understand. Some algorithms that succeeded in continuous domain were extended specifically to deal with binary search spaces, including a discrete differential evolution (Kafafy, Bounekkar, & Bonnevay, 2012) and a binary cuckoo search (Layeb, Lahouesna, & Kireche, 2013).

Numerous studies investigated performance improvement strategies for binary genetic algorithms, which are popular and can be applied in a straightforward way to solve MOKPs. Aghezzaf and Naimi (2009) improved the recombination procedure by introducing a two-stage crossover. In the first stage, the offspring is initialized in the way the similar genes of parents are inherited. In the second stage, the remaining genes are selected from parents based on a fitness information. Disruptive effects of crossover on MOKPs often manifest in the loss of diversity. Ishibuchi et al. (2010a) showed that the genetic diversity can be lost since traditional crossovers always generate offspring in the segment between two parents. A nongeometric binary crossover was proposed to produce offspring outside that segment. As a result, a significant improvement in the diversity of solutions was observed, without the convergence being deteriorated. Sato, Aguirre, and Tanaka (2013) suggested to limit a number of parent genes that are exchanged during recombination. Ishibuchi, Tanigaki, Masuda, and Nojima (2014) further examined this idea controlling the distance to offspring by a user-defined parameter during recombination. Good results for a small parent-offspring distance were obtained due to the increase in diversity. However, the approach slows down the convergence on many-objective instances. The major shortcoming of such approaches lies in the need to specify parameters introduced for controlling the parent-offspring distance and the restricted mating.

Similarly to single-objective optimization, local search and hybridization strategies proved effective in improving the performance of MOEAs. Knowles and Corne (2000) showed that memetic algorithms exhibit better performance than a baseline algorithm on MOKPs. Li, Zhang, Tsang, and Ford (2004) improved the results presented by Jaszkiewicz (2002) using a local search strategy in the estimation of distribution algorithm. Such approaches aim to reap advantages of different search strategies and have downsides related to the increased complexity and the need to balance local and global search.

Aiming at improving the scalability and the effectiveness of search, this study proposes solving MOKPs by neuroevolution in an attempt to reduce the size of the search space and to benefit from the learning capability of neural networks. Neuroevolution refers to the application of evolutionary algorithms to evolve neural networks (Floreano, Dürr, & Mattiussi, 2008). Neuroevolution is represented by a variety of algorithms (Yao, 1999) that are often categorized by the type of encoding. Direct encodings translate each component of the genome to a specific part of the neural network (Stanley & Miikkulainen, 2002). They are simple to understand and can be used with traditional variation operators. Their major disadvantage is that the length of chromosome grows with the size of network. Indirect encodings attempt to overcome this limitation by using genes multiple times (Stanley, D'Ambrosio, & Gauci, 2009). The reuse of genes takes its inspiration from the development of embryos in nature (Stanley & Miikkulainen, 2003). Although indirect encodings provide the potential to evolve large neural networks, they are more complex and may necessitate specifically designed operators. This study uses a direct encoding of the neural network with a real number representation. Heidrich-Meisner and Igel (2009) showed that such type of neuroevolution can be advantageous.

3. Multiobjective knapsack problem

Zitzler and Thiele (1999) originally suggested 0-1 multiobjective knapsack problem for benchmarking the performance of EMOAs. A general single objective knapsack problem was extended by allowing an arbitrary number of knapsacks. Formally, this problem can be defined as follows.

$$\begin{aligned} \underset{\boldsymbol{x}\in\{0,1\}^n}{\text{maximize}} \quad & f_j(\boldsymbol{x}) = \sum_{i=1}^{n} x_i p_{ij} \\ \text{subject to} \quad & \sum_{i=1}^{n} x_i w_{ij} \le c_j \qquad \forall j \in \{1,\dots,m\} \end{aligned} \tag{1}$$

where

p_{ij} is profit of the i-th item w.r.t. the j-th knapsack
w_{ij} is weight of the i-th item w.r.t. the j-th knapsack
c_j is capacity of the j-th knapsack

m and n are the number of knapsacks and items, respectively.

A solution to the problem is encoded as a binary string $\boldsymbol{x}$ such that $\forall i \in \{1,\dots,n\}$:

$$x_i = \begin{cases} 1 & \text{if the } i\text{-th item is selected} \\ 0 & \text{otherwise.} \end{cases}$$

An instance of MOKP can be defined by randomly generating integer values for the profits and weights. Using a feasibility ration r_j, the capacity of the j-th knapsack can be determined by

$$c_j = r_j \sum_{i=1}^{n} w_{ij}. \tag{2}$$

From the mathematical formulation, it can be understood that the simultaneous maximization of profits in different knapsacks is in general conflicting. As a consequence, a set of Pareto optimal solutions is expected to the problem defined in (1).

4. Concepts

This section outlines key concepts necessary for the subsequent discussion in the paper. We use the notation where $\boldsymbol{x}$ is the decision vector, $\Omega \subseteq \mathbb{R}^n$ is the feasible decision (or solution) space and $\boldsymbol{f}(\boldsymbol{x})$ is the objective vector defined in the objective space $\mathbb{R}^m$. The MOKP formulation given in (1) implies the maximization of the objective functions. For convenience, in the following, the optimization problem is considered in terms of minimization by changing the sign of the objectives to negative.

Since multiple objectives are simultaneously optimized, solutions are partially ordered in the objective space. Under the given circumstances, the comparison of different solutions can be performed on the basis of the Pareto dominance relation.

For two solutions $\boldsymbol{a}$ and $\boldsymbol{b}$ from Ω, a solution $\boldsymbol{a}$ is said to dominate a solution $\boldsymbol{b}$ (denoted by $\boldsymbol{a} \prec \boldsymbol{b}$) if

$$\forall i \in \{1,\dots,m\} : f_i(\boldsymbol{a}) \le f_i(\boldsymbol{b}) \wedge \exists j \in \{1,\dots,m\} : f_j(\boldsymbol{a}) < f_j(\boldsymbol{b}). \tag{3}$$

The presence of multiple conflicting objectives gives rise to a set of optimal solutions. The concepts of optimality for multiobjective optimization are defined as follows.

A solution $\boldsymbol{x}^* \in \Omega$ is Pareto optimal if and only if

$$\nexists \boldsymbol{y} \in \Omega : \boldsymbol{y} \prec \boldsymbol{x}^*. \tag{4}$$

For a multiobjective optimization problem, the Pareto optimal set (or Pareto set, for short) is defined as

$$\mathcal{PS}^* = \{\boldsymbol{x}^* \in \Omega \mid \nexists \boldsymbol{y} \in \Omega : \boldsymbol{y} \prec \boldsymbol{x}^*\}. \tag{5}$$

For a multiobjective optimization problem and the Pareto optimal set $\mathcal{PS}^*$, the Pareto optimal front (or Pareto front, for short) is defined as

$$\mathcal{PF}^* = \{\boldsymbol{f}(\boldsymbol{x}^*) \in \mathbb{R}^m \mid \boldsymbol{x}^* \in \mathcal{PS}^*\}. \tag{6}$$

The outcome of a multiobjective optimization algorithm is considered to be a set of nondominated solutions while all obtained dominated solutions are discarded being of no interest due to their inferiority with respect to the Pareto dominance relation. This is formalized by the notion of an approximation set (Zitzler, Thiele, Laumanns, Fonseca, & Grunert da Fonseca, 2003).

A set of objective vectors $A \subseteq \mathbb{R}^m$ is called an approximation set if any element of A does not dominate any other objective vector in A.

The comparison of multiobjective optimization algorithms is typically conducted assessing the quality of produced approximation sets. For this purpose, quality indicators can be used. They generally map approximation sets to the set of real numbers. This is a useful feature that allows for the application of statistical testing procedures. Following suggestions of Knowles, Thiele, and Zitzler (2006), we use the Pareto compliant quality indicators.

The epsilon indicator is based on the concept of additive ϵ-dominance (Zitzler et al., 2003). It gives the minimum factor ϵ such that any objective vector in a reference set R is ϵ-dominated by at least one objective vector in A

$$I_\epsilon = \inf_{\epsilon\in\mathbb{R}} \{\forall \mathbf{r} \in R \; \exists \mathbf{a} \in A : \; \mathbf{a} \preceq_\epsilon \mathbf{r}\}. \tag{7}$$

This quality indicator mainly assesses the convergence of A, with smaller values of I_ϵ being preferable.

The hypervolume indicator (Zitzler & Thiele, 1998b), also referred to as $\mathcal{S}$ metric, measures the volume of the objective space that is dominated by an approximation set and is bounded by a reference point. It can be defined as the Lebesgue measure Λ of the union of hypercuboids in the objective space as

$$I_H = \Lambda\left(\bigcup_{\boldsymbol{a}\in A} \{f_1(\boldsymbol{a}'),\dots,f_m(\boldsymbol{a}') : \boldsymbol{a} \prec \boldsymbol{a}' \prec \boldsymbol{r}\}\right) \tag{8}$$

where $A = \{\boldsymbol{a}_1,\dots,\boldsymbol{a}_{|A|}\}$ is an approximation set and $\boldsymbol{r}$ is an appropriately chosen reference point. This quality indicator can measure both the convergence and diversity of A, with higher values of I_H being preferable.

In this work, we develop neuroevolutionary multiobjective optimization algorithm for solving instances of MOKPs defined by (1). Neuroevolution is the optimization of neural networks by using evolutionary computation techniques. It offers a way for both learning the parameters and determining the optimal topology of neural networks.

Using a representation, an optimization problem can be separated into a genotype-phenotype mapping and phenotype-fitness mapping (Rothlauf, 2006). The notions of genotype and phenotype are borrowed from nature. The genotype space Φ_g defines a search space where stochastic evolutionary operators act during reproduction. A solution representation in this space is given by a chromosome of the length l. The phenotype space Φ_p represents a space of actual solutions to the problem at hand. The phenotype representation is used to compute the objective values for the given solution. The evaluation of a given individual can be expressed as a composite function $f_p \circ f_g$, where the function

$$f_g : \Phi_g \longmapsto \Phi_p \tag{9}$$

performs the genotype-phenotype mapping with Φ_p corresponding to the decision space $\mathbb{R}^n$ and the function

$$f_p : \Phi_p \longmapsto \Phi_f \tag{10}$$

performs the phenotype-fitness mapping with Φ_f being equivalent to the objective space $\mathbb{R}^m$.

Table 1
Results for evolutionary and neuroevolutionary approaches. The values refer to median and interquartile range of epsilon (eps) and hypervolume (hv) indicators. Best performance is highlighted with gray background. The symbol † indicates a statistical difference between the respective and best performing algorithm.

2 knapsacks		One-point crossover	Two-point crossover	Uniform crossover	Neuroevolution
500 items	eps	4.26e+00 (1.7e-02)†	4.26e+00 (1.8e-02)†	4.25e+00 (1.6e-02)†	4.10e+00 (1.1e-02)
	hv	5.92e-01 (2.3e-02)†	6.02e-01 (1.4e-02)†	6.25e-01 (2.0e-02)†	7.91e-01 (8.9e-03)
1000 items	eps	4.13e+00 (2.1e-02)†	4.13e+00 (1.7e-02)†	4.12e+00 (1.1e-02)†	3.88e+00 (1.4e-02)
	hv	5.72e-01 (1.8e-02)†	5.81e-01 (1.4e-02)†	5.96e-01 (1.1e-02)†	8.35e-01 (6.1e-03)
2000 items	eps	4.26e+00 (1.8e-02)†	4.25e+00 (2.1e-02)†	4.23e+00 (1.6e-02)†	3.97e+00 (1.0e-02)
	hv	4.76e-01 (9.3e-03)†	4.88e-01 (1.0e-02)†	5.15e-01 (1.0e-02)†	8.03e-01 (1.0e-02)
5000 items	eps	4.13e+00 (1.0e-02)†	4.11e+00 (1.4e-02)†	4.06e+00 (7.1e-03)†	3.76e+00 (1.4e-02)
	hv	4.40e-01 (4.7e-03)†	4.62e-01 (9.5e-03)†	5.21e-01 (5.0e-03)†	8.54e-01 (5.7e-03)
10000 items	eps	4.62e+00 (8.9e-03)†	4.58e+00 (1.2e-02)†	4.44e+00 (6.9e-03)†	4.06e+00 (1.8e-02)
	hv	2.28e-01 (6.2e-03)†	2.62e-01 (7.3e-03)†	4.11e-01 (5.2e-03)†	8.24e-01 (8.7e-03)
			3 knapsacks		
500 items	eps	3.42e+00 (2.4e-02)†	3.42e+00 (1.7e-02)†	3.42e+00 (1.9e-02)†	3.22e+00 (3.0e-02)
	hv	4.07e-01 (1.2e-02)†	4.19e-01 (1.4e-02)†	4.21e-01 (9.8e-03)†	6.33e-01 (1.5e-02)
1000 items	eps	3.71e+00 (1.1e-02)†	3.71e+00 (2.3e-02)†	3.70e+00 (2.0e-02)†	3.41e+00 (3.0e-02)
	hv	3.25e-01 (9.2e-03)†	3.32e-01 (8.3e-03)†	3.46e-01 (4.2e-03)†	6.18e-01 (2.4e-02)
2000 items	eps	3.68e+00 (1.6e-02)†	3.68e+00 (1.5e-02)†	3.65e+00 (1.3e-02)†	3.33e+00 (3.2e-02)
	hv	2.62e-01 (7.9e-03)†	2.71e-01 (9.7e-03)†	2.98e-01 (5.8e-03)†	6.27e-01 (2.0e-02)
5000 items	eps	3.53e+00 (9.2e-03)†	3.51e+00 (1.2e-02)†	3.47e+00 (8.8e-03)†	3.15e+00 (1.8e-02)
	hv	1.87e-01 (6.6e-03)†	2.03e-01 (4.1e-03)†	2.51e-01 (4.1e-03)†	6.29e-01 (1.3e-02)
10000 items	eps	3.81e+00 (1.1e-02)†	3.78e+00 (1.2e-02)†	3.69e+00 (8.4e-03)†	3.30e+00 (4.0e-02)
	hv	1.08e-01 (3.7e-03)†	1.23e-01 (4.8e-03)†	1.97e-01 (4.7e-03)†	6.21e-01 (2.6e-02)
			4 knapsacks		
500 items	eps	3.07e+00 (1.7e-02)†	3.07e+00 (1.9e-02)†	3.06e+00 (1.6e-02)†	2.82e+00 (2.7e-02)
	hv	2.82e-01 (7.8e-03)†	2.87e-01 (6.1e-03)†	2.99e-01 (5.4e-03)†	4.93e-01 (2.3e-02)
1000 items	eps	3.28e+00 (1.2e-02)†	3.27e+00 (2.2e-02)†	3.27e+00 (1.8e-02)†	3.00e+00 (3.5e-02)
	hv	2.08e-01 (3.3e-03)†	2.13e-01 (9.7e-03)†	2.33e-01 (5.4e-03)†	4.62e-01 (2.2e-02)
2000 items	eps	3.16e+00 (1.1e-02)†	3.16e+00 (8.5e-03)†	3.13e+00 (1.1e-02)†	2.85e+00 (1.7e-02)
	hv	1.53e-01 (5.6e-03)†	1.58e-01 (1.9e-03)†	1.83e-01 (3.2e-03)†	4.59e-01 (2.7e-02)
5000 items	eps	3.34e+00 (1.7e-02)†	3.33e+00 (1.1e-02)†	3.29e+00 (3.8e-03)†	2.96e+00 (1.9e-02)
	hv	9.63e-02 (4.5e-03)†	1.05e-01 (2.6e-03)†	1.36e-01 (2.2e-03)†	4.54e-01 (2.6e-02)
10000 items	eps	3.51e+00 (9.0e-03)†	3.50e+00 (6.9e-03)†	3.42e+00 (1.1e-02)†	3.04e+00 (3.0e-02)
	hv	5.01e-02 (2.4e-03)†	5.76e-02 (1.7e-03)†	9.94e-02 (2.2e-03)†	4.40e-01 (1.6e-02)
			5 knapsacks		
500 items	eps	3.18e+00 (2.3e-02)†	3.18e+00 (2.4e-02)†	3.16e+00 (1.7e-02)†	2.96e+00 (3.0e-02)
	hv	1.70e-01 (4.7e-03)†	1.74e-01 (7.8e-03)†	1.81e-01 (4.7e-03)†	3.19e-01 (2.3e-02)
1000 items	eps	3.01e+00 (1.2e-02)†	3.00e+00 (8.9e-03)†	2.99e+00 (1.3e-02)†	2.77e+00 (2.9e-02)
	hv	1.21e-01 (6.8e-03)†	1.27e-01 (4.0e-03)†	1.40e-01 (3.6e-03)†	3.22e-01 (2.5e-02)
2000 items	eps	3.17e+00 (6.5e-03)†	3.16e+00 (1.0e-02)†	3.14e+00 (1.2e-02)†	2.87e+00 (5.2e-02)
	hv	8.53e-02 (1.6e-03)†	9.08e-02 (3.6e-03)†	1.10e-01 (4.1e-03)†	3.22e-01 (2.2e-02)
5000 items	eps	3.18e+00 (1.1e-02)†	3.17e+00 (1.0e-02)†	3.13e+00 (8.7e-03)†	2.85e+00 (5.2e-02)
	hv	4.95e-02 (2.5e-03)†	5.42e-02 (2.3e-03)†	7.47e-02 (1.7e-03)†	3.08e-01 (1.6e-02)
10000 items	eps	3.37e+00 (1.1e-02)†	3.36e+00 (1.0e-02)†	3.29e+00 (8.9e-03)†	2.93e+00 (3.9e-02)
	hv	2.37e-02 (9.8e-04)†	2.71e-02 (9.5e-04)†	5.01e-02 (1.1e-03)†	3.03e-01 (2.4e-02)

5. Neuroevolution Multiobjective Optimization

An approach developed in our study uses the framework of $\mathcal{S}$ metric selection evolutionary multiobjective optimization algorithm (SMS-EMOA) proposed by Beume et al. (2007). In SMS-EMOA, the population undergoes a steady-state evolutionary process where a single offspring is produced in each generation. The mating selection is performed by picking up a set of different population members uniformly at random. The variation procedure generates a single offspring by recombining selected parent individuals. The environmental selection procedure updates the population by removing an individual with the smallest hypervolume contribution in the last nondominated front. The evolution goes on for a user specified number of generations.

The main idea of the proposed $\mathcal{S}$ metric selection neuroevolutionary multiobjective optimization algorithm (SMS-NEMOA) for solving MOKPs consists of generating solutions by artificial neural networks (ANNs). This is implemented in the variation procedure. ANNs are computational models that attempt to mimic the structure and function of the human brain. In ANNs, neurons are the basic information processing units that communicate with each other through weighted connections. By modifying the strength of the connections, memory and learning can be created. Thus, the rationale behind our approach is to exploit these features in order to surpass the search process.

We use a feedforward neural network with one hidden layer. The standard sigmoid function is used as the activation function in all hidden and output neurons $\sigma(z) = 1/(1 + e^{(-z)})$. The use of sigmoid activation functions in hidden neurons is intended to account for nonlinearities. The sigmoid function with appropriate threshold in the output layer allows to obtain a binary value that represents a decision regarding the selection of a specific item.

In order to achieve a desired behavior, the parameters of neural network must be adjusted. This is known as a learning process, which is commonly performed by a gradient-based optimization aiming at minimizing the difference between the network outputs and target values. Although the realization of the herein explored idea is based on the ability of neural networks to perform classification, there is an important distinction between a traditional classification task and the one employed in the context of solving MOKPs. In the former, the learning is performed based on a dataset with known input and target variables. In the latter, there is no such data and a gradient-based learning cannot be applied.

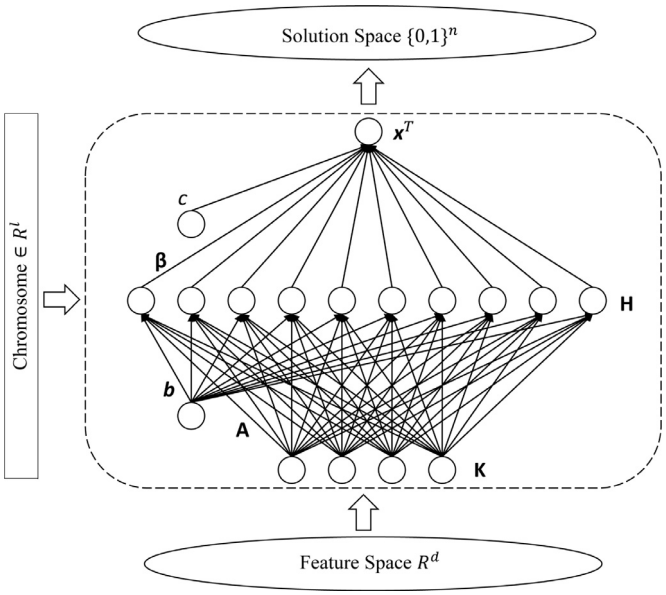

Fig. 1. Depiction of the proposed genotype-phenotype mapping. The neural network is applied to all items in the feature space, generating a complete solution in $\{0, 1\}^n$. This is illustrated in the matrix form.

Another important aspect is that the neural network performance must be evaluated accounting for multiple objectives.

We use neuroevolution because it provides the potential to overcome limitations associated with traditional learning methods and to evolve the genotype-phenotype mapping f_g. The letter represents the major difference with traditional evolutionary computation approaches that seek to evolve the solutions itself when solving MOKPs. Neuroevolution is incorporated into the framework of SMS-EMOA by implementing appropriate encoding, reproduction and evaluation schemes. We use a direct encoding, which means the genes composing the chromosome straightforwardly correspond to the parameters of neural network. Each chromosome is represented by a real-valued string. Depending on the location, the genes encode weights, biases and a boolean mask for the hidden layer. Each component of the mask is determined by checking the sign of the corresponding gene. The false values indicate the neurons with all incoming and outgoing connections to be removed from the neural network. This is a simple yet effective encoding scheme that can learn the parameters and topology of neural network. It reduces a human labor as the user only needs to specify the maximum number of hidden neurons while the optimal topology is determined by evolution. It is also appropriate for the application of traditional reproduction operators for continuous optimization. We investigate the effects of different operators in the experimental study.

As to evaluation, a suitable function f_g performing the genotype-phenotype mapping is defined as follows. First, the items associated with the MOKP at hand are placed into a d-dimensional feature space, where each item is represented by the vector $\boldsymbol{\phi}$. Then, the chromosome is decoded into the neural network with weights $\boldsymbol{A}$ and biases $\boldsymbol{b}$ in the hidden layer and weights $\boldsymbol{\beta}$ and bias c in the output layer. Next, the coordinates of each item in the feature space are fed into the neural network. For each feature vector, the output of neural network gives the component of a solution to MOKP. The overall genotype-phenotype mapping can be defined as

$$f_g = o \circ h \circ \phi. \tag{11}$$

In the above equation, the function ϕ places the items into the feature space. Some possible choices for ϕ are discussed in the experimental study. All the items stored in the rows of matrix $\boldsymbol{K} \in \mathbb{R}^{n\times d}$ are first mapped into the hidden layer by $h = \sigma(\boldsymbol{K}\boldsymbol{A} + \boldsymbol{1}\boldsymbol{b}^{\mathrm{T}})$ and then mapped to the output layer by $o = round(\sigma(\boldsymbol{H}\boldsymbol{\beta} + \boldsymbol{1}c))$, where $\boldsymbol{H}$ is the representation of $\boldsymbol{K}$ in the hidden layer and σ is the sigmoid function. The outcome of f_g determines a solution $\boldsymbol{x}^{\mathrm{T}}$ whose objective values are calculated using the MOKP formulation, which concludes the evaluation procedure.

Fig. 1 graphically illustrates the idea of the proposed genotype-phenotype mapping. It is shown that the chromosome represented by the vector of real numbers is transformed into the neural network characterized by the weights $\boldsymbol{A}$ and $\boldsymbol{\beta}$ as well as the biases $\boldsymbol{b}$ and c. All the items in the feature space, which are stored in the rows of the matrix $\boldsymbol{K}$, are propagated through the neural network, yielding a nonlinear representation in the hidden layer $\boldsymbol{H}$ and a solution to the MOKP in the output layer $\boldsymbol{x}^{\mathrm{T}}$. The process of propagating different items can be parallelized, thereby allowing for the scalability of the proposed approach.

6. Computational experiments

This section discusses the computational experiments performed to validate the proposed approach. First, it is benchmarked against state-of-the-art approaches. Then, its different variants are investigated.

6.1. Experimental setup

The experimental study involves MOKP instances having between 2 and 5 knapsacks with items ranging from 500 to 10000. Each instance is defined by randomly generated the values of profits and weights in the interval [10,100]. Similarly to Zitzler and Thiele (1999), the feasibility ratio of $r_j = 0.5 \;\; \forall j \in \{1, \ldots, m\}$ is used. Test problems are denoted in the form MOKP_m_n. As an example, consider MOKP_2_500 that indicates the MOKP with 2 knapsacks and 500 items.

In multiobjective optimization, there are the decision and the objective search spaces. In EMOAs, specific operators are implemented to perform the search in each of these spaces. Meaningful insights can be obtained by studying the effects of different strategies one at a time. This also ensures the fairness of their comparison. Therefore, all the algorithms used in the experiments share the framework outline in Algorithm 1, with the mating and environmental selection procedures corresponding to SMS-EMOA (Beume et al., 2007). As the main contribution of our study is a scheme for generating solutions implemented in the variation procedure, we compare it with state-of-the-art recombination operators based on a binary encoding. For better readability, we only use EA and NEA for respectively referring to traditional evolutionary and neuroevolutionary variant of $\mathcal{S}$ metric selection multiobjective optimization algorithm.

Due to the stochastic nature, 21 independent runs with different random number initializations were performed by the algorithms on each test problem. The odd number of runs was used for an accurate estimation of median. To study the scalability, for all test runs the population size and the maximum number of generations were set to 100 and 1000, respectively.

For computing quality indicators, a reference set is constructed by combining all nondominated solutions obtained in the experiments. All objective values are normalized using the minimum and maximum objective values in the reference set. The resulting reference set is used as R for calculating I_ϵ in (7). The vector of $\boldsymbol{1}$ is used for computing I_H in (8).

For statistically sound conclusions, a statistical analysis of the results was performed (Derrac, García, Molina, & Herrera, 2011). For each problem, a pairwise testing was carried out to determine whether the observed difference in the performance is due

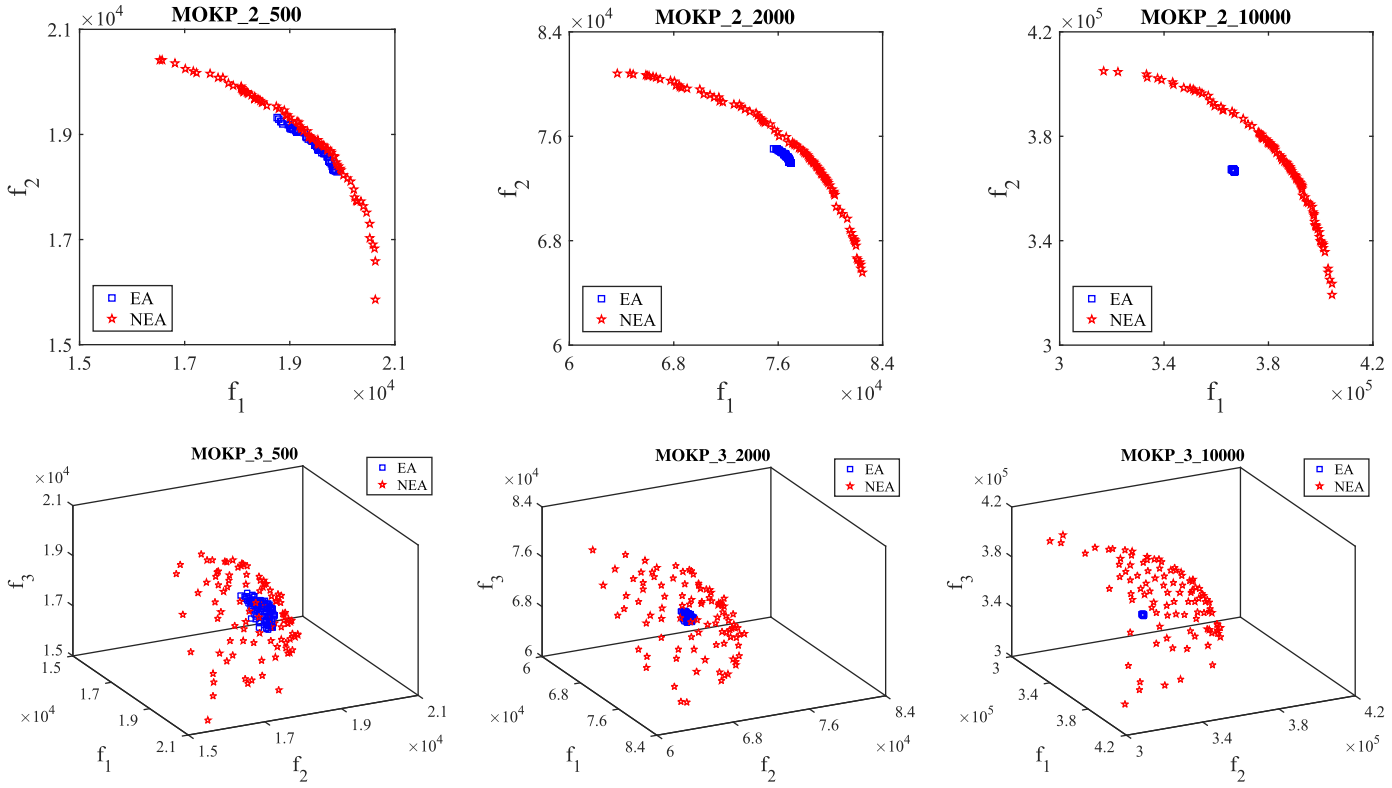

Fig. 2. Two and three-dimensional Pareto front approximations obtained by evolutionary (EA) and neuroevolutionary (NEA) approaches. The plots show the results of the run with the median hypervolume value.

to chance. To this end, a nonparametric Wilcoxon rank sum test was applied at a significance level of $\alpha = 0.05$.

6.2. Comparative study

First, we investigate the performance of the proposed neuroevolutionary approach (NEA) in comparison with the evolutionary approach (EA). Both approaches operate identically in the objective space and differently in the decision space. EA evolves binary strings that represent the solutions to MOKP, whereas NEA evolves neural networks for generating these solutions. EA is represented by state-of-the-art crossover operators such as one-point, two-point and uniform crossovers. These are used with the crossover probability of 0.8 and the mutation of $2 \times 1/l$ where l is the chromosome length. Such settings are frequently used when handling MOKPs (Ishibuchi et al., 2010a; 2015; Ishibuchi et al., 2014). For these experiments, NEA is adopted with a real-coded genetic algorithm using a simulated binary crossover (SBX) and polynomial mutation (PM). SBX is chosen because it attempts to operate in a similar way one-point crossover does for binary representations (Deb & Agrawal, 1995). In the following section, the effects of different variation operators on the performance of NEA are examined.

Table 1 shows the results obtained by both approaches. The dominant performance of NEA can be readily observed. It gives significantly better results with respect to both quality indicators on all considered MOKP instances. This is suggested by small values of I_ϵ and large values of I_H as well as the results of statistical tests. Although the values of quality indicators provide clear insights about the relative performance of the approaches, the cause of the observed behavior cannot be fully appreciated. This can be better understood by visually analyzing Pareto front approximations.

Figs. 2 and 3 visualize the obtained results for problems with 500, 2000 and 10000 items. For two and three objective instances, scatter plots are shown in Fig. 2. For higher dimensions, parallel coordinates are used in Fig. 3. The plots depict approximation sets having the median hypervolume value. The results for EA with uniform crossover are shown, as it gives the best results among traditional crossover operators. The plots clearly indicate the superiority of the proposed NEA. The obtained solutions exhibit both better convergence and diversity. With regard to diversity, the solutions from NEA cover a large portion of the objective space which contrast with those obtained by EA that are located in a small region. This can be easily observed in scatter and parallel coordinate plots. An important observation is that the difference in results between the two approaches becomes larger when the dimensionality of MOKP increases. When considering results of EA, it is evident that diversity and convergence degrade when the number of items grows. This appears to be the case for all considered dimensions of the objective space.

To quantitatively illustrate that the difference between the results of the two approaches becomes larger with increasing dimensionality, we calculate the relative improvement in the epsilon and hypervolume indicators (ΔI_ϵ and ΔI_H) as

$$\begin{aligned} \Delta I_\epsilon &= |I_\epsilon^1 - I_\epsilon^2| / I_\epsilon^1 \\ \Delta I_H &= |I_H^1 - I_H^2| / I_H^1 \end{aligned} \tag{12}$$

where I_ϵ^i and I_H^i are the values of the corresponding indicators for the i-th approach ($i = \{1, 2\}$).

Fig. 4 plots the values of relative improvements in quality indicators against the number of items in MOKPs. In this figure, a general trend for both indicators can be readily observed. The factor by which NEA outperforms EA becomes increasingly larger for a higher number of items.

Nevertheless, the values of ΔI_ϵ are much smaller than those of ΔI_H and do not grow considerably with the increase in dimensionality. This is because I_ϵ mainly assesses the convergence, as it can be seen in Figs. 2 and 3 the results of EA and NEA differ the most

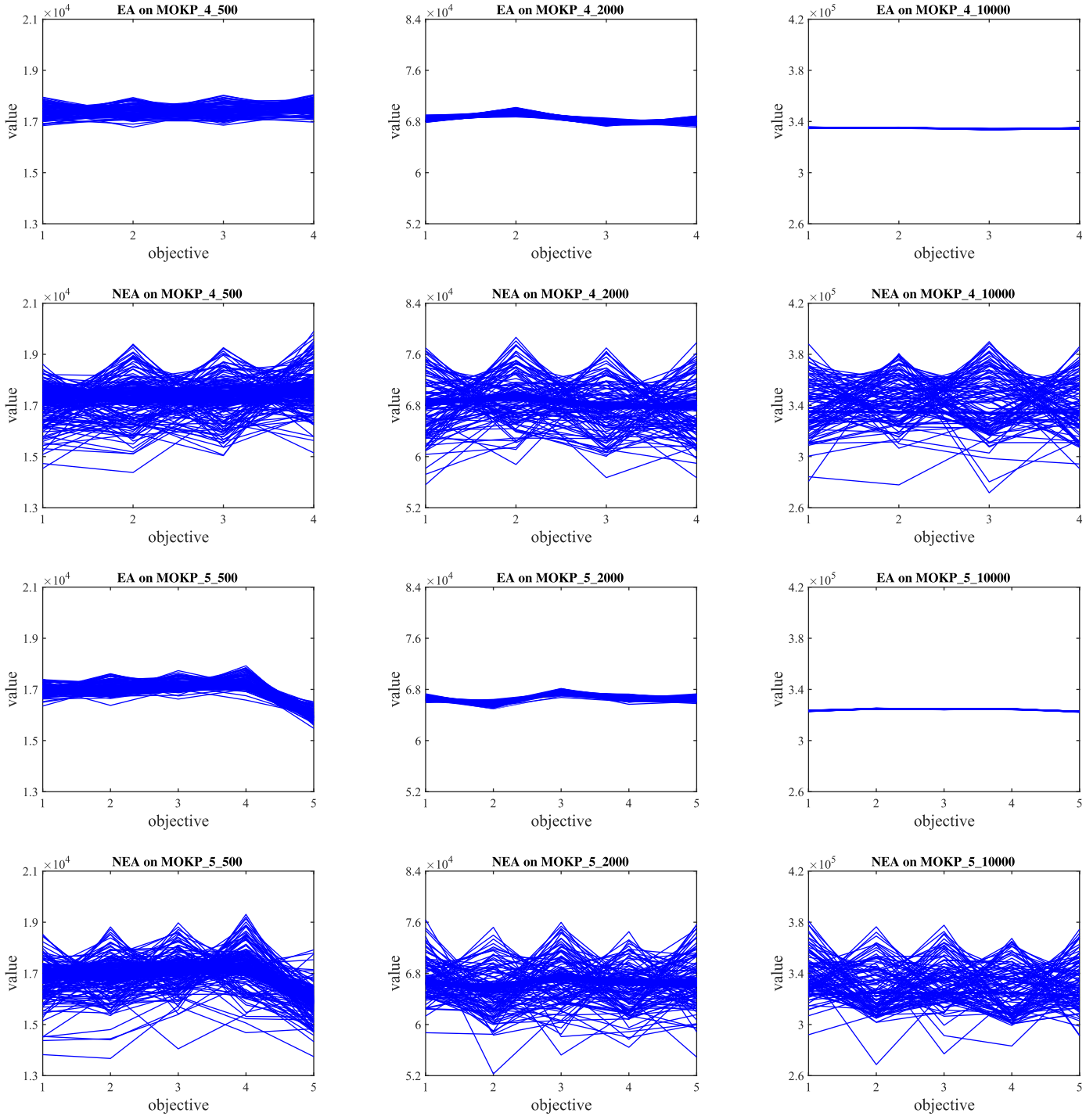

Fig. 3. Four and five-dimensional Pareto front approximations obtained by evolutionary (EA) and neuroevolutionary (NEA) approaches. The plots show the results of the run with the median hypervolume value.

with respect to the diversity. This feature is suitably captured by the hypervolume.

For instance, the difference between I_H of EA and NEA on the MOKP with 2 knapsacks and 10000 items is nearly equal to the total I_H of EA. For MOKP with 5 knapsacks and 10000 items, this difference is approximately 5 times. Such a dramatic discrepancy in the performance can be attributed to shortcomings of traditional binary crossovers. The deficiency of crossover operators grows when the size of chromosome increases. As discussed in Section 2, this issue is often addressed by restricting mating pool so that similar parents undergo recombination. Alternatively, the diversity of solutions can be ensured by a suitable selection procedure in the objective space. Our results demonstrate this issue can be also effectively addressed by neuroevolution.

The superior performance can be explained by important features of the proposed NEA. On one hand, the scalability stems from the fact that the size of the search space is determined by the number of connections in the neural network and not by the number of items, as opposed to traditional approaches. Thus, the length of solution to a MOKP can grow while the search space needed to

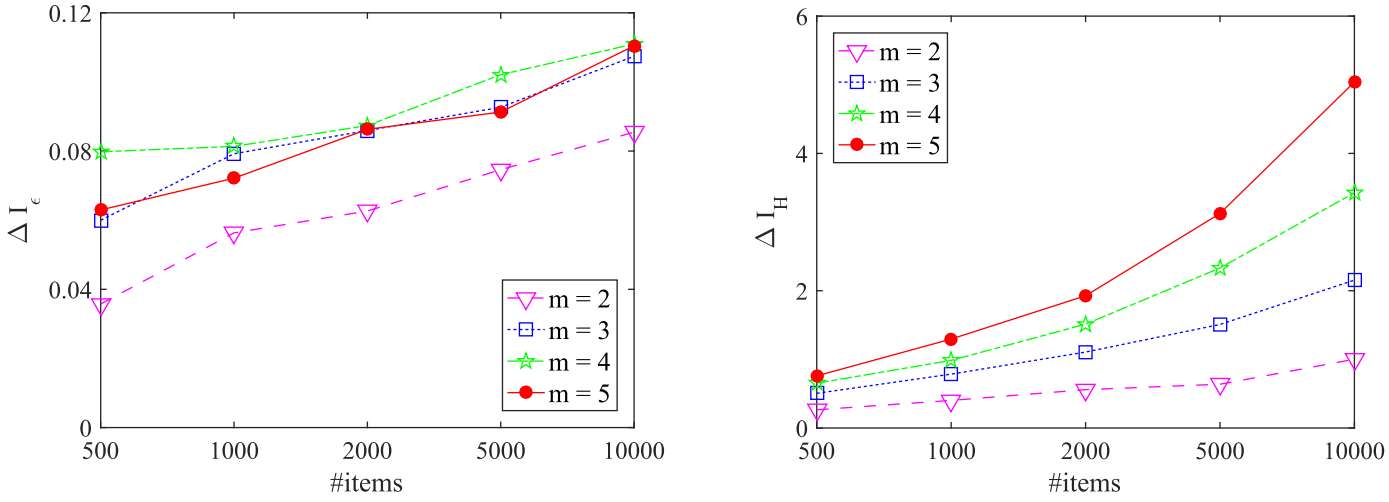

Fig. 4. Improvement in indicator values depending on the number of items.

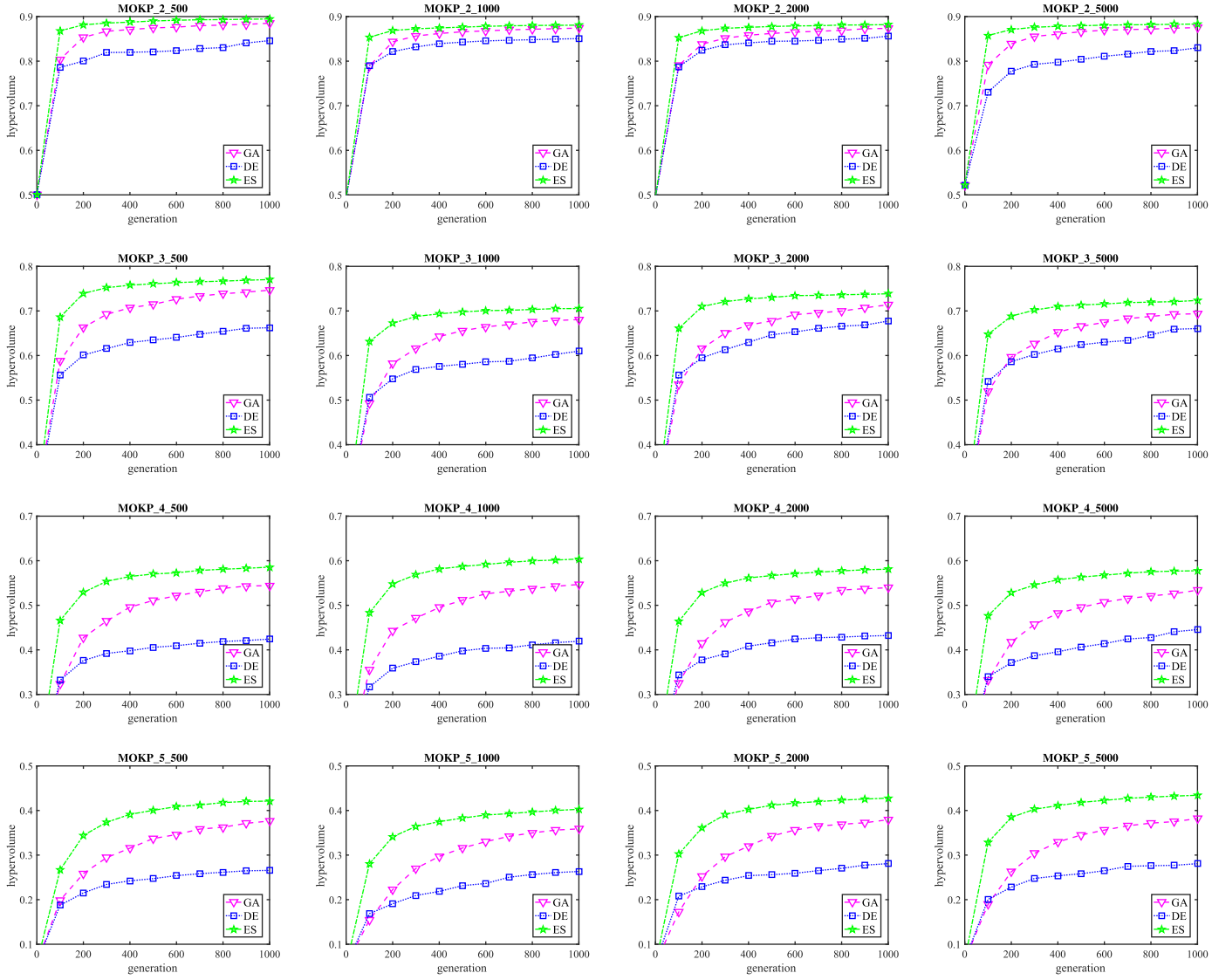

Fig. 5. Evolution of the hypervolume indicator for different variation operators. The plots refer to the median values. The higher the better.

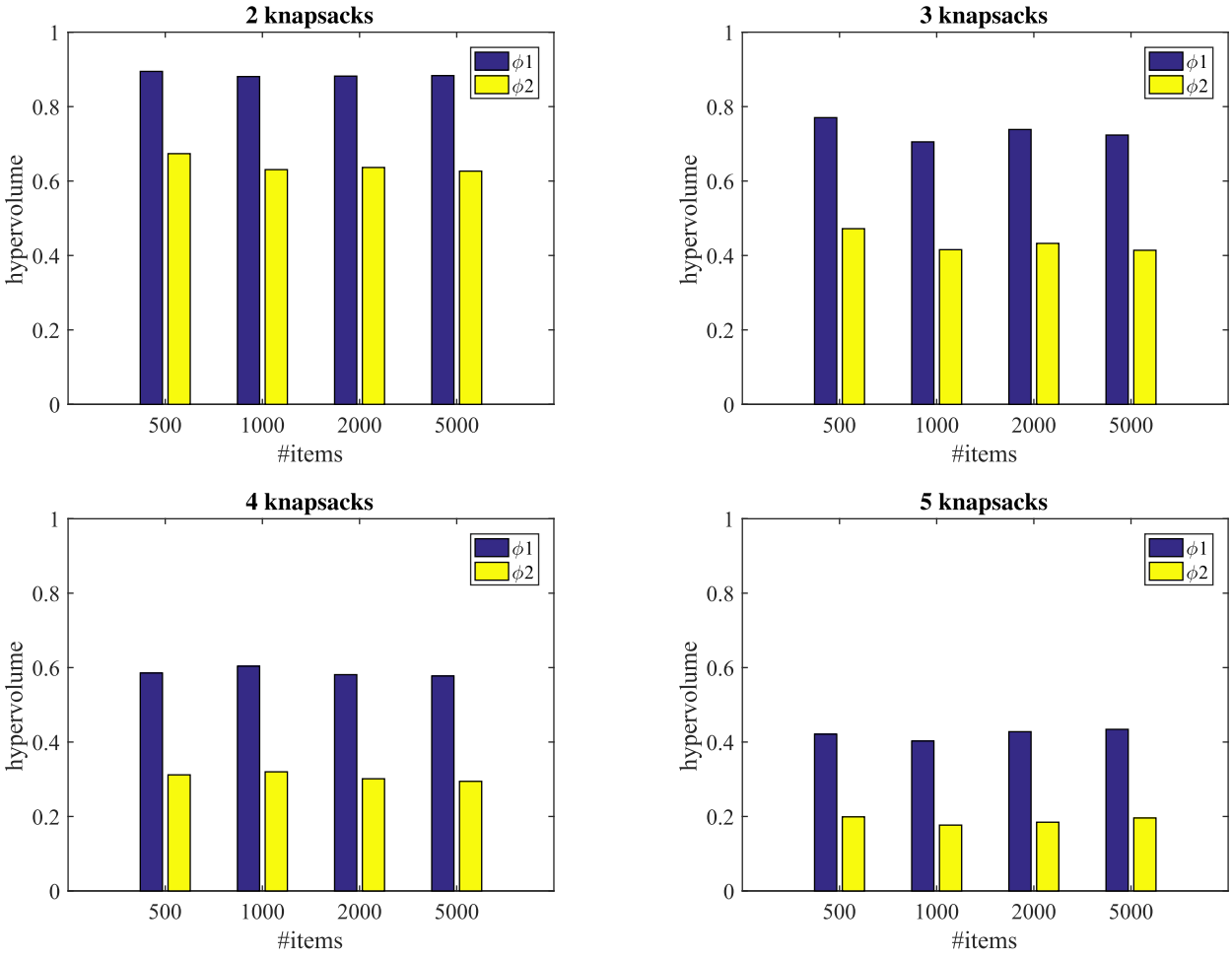

Fig. 6. Performance of neuroevolution with different feature mappings.

be explored by NEA can be the same. This is appealing to practical applications where instances of MOKPs often arise as large scale problems. On the other hand, the learning capability of neural network is exploited, as the knowledge about all items is taken into account when the network decides on a particular item. This knowledge is accumulated during the evolution and is stored in its weights and biases. Obviously, the proposed scheme is more complex in terms of implementation and computing overhead due to matrix multiplications, which can be regarded as its major disadvantage.

6.3. Effect of variation operator

The way in which the new individuals are generated influences significantly the effectiveness of the search process. In neuroevolution, this is performed by the variation operator. This operator produces offspring by stochastically manipulating the chromosomes of parent individuals, with several possibilities available to realize this process. One of the advantages of the used real encoding is its ability to adapt different variation operators proved effective in algorithms for continuous search spaces. The ability to use different search paradigms is important because there is no single strategy that works the best for all the cases. Each one has its own characteristics and can be useful in different situations. Although some recommendations for the choice of a particular strategy can be found, in practice for the given problem this issue is often addressed by experimentation.

Table 2
Parameter settings (l - is the chromosome length).

Operator	Parameters
SBX	$p_c = 1,\ \eta_c = 20$
PM	$p_m = 1/l,\ \eta_m = 20$
DE	$CR = 1,\ F = 0.5$
ES	$\tau_0 = 1/\sqrt{2l},\ \tau_1 = 1/\sqrt{2\sqrt{l}},\ \boldsymbol{\sigma}_0 = 1/\sqrt{1/(3l)}$

Given its importance to the overall performance of neuroevolution, we investigate the effects of three popular variation operators. The first is a real-coded genetic algorithm (GA) operator, which relies on simulated binary crossover (SBX) and polynomial mutation (PM) (Deb, 2001). The second is differential evolution (DE) operator with rand/1/bin variant (Storn & Price, 1997) and PM. The third is evolution strategy (ES) operator, which is used with a non-isotropic mutation (Beyer & Schwefel, 2002). The parameter settings for the three operators are shown in Table 2.

Fig. 5 depicts the evolution of the hypervolume for the three different variation operators. For MOKP instances with different numbers of knapsacks and items, a common trend can be readily observed. The best performing variant is the one using ES operator. The second and third performance is provided by GA and DE, respectively. Thus, these results show that the performance of neuroevolution discussed in the previous section can be improved by using an appropriate scheme for producing offspring.

Both GA and ES treat the genes in chromosome independently when generating offspring. For this purpose, GA recombines a

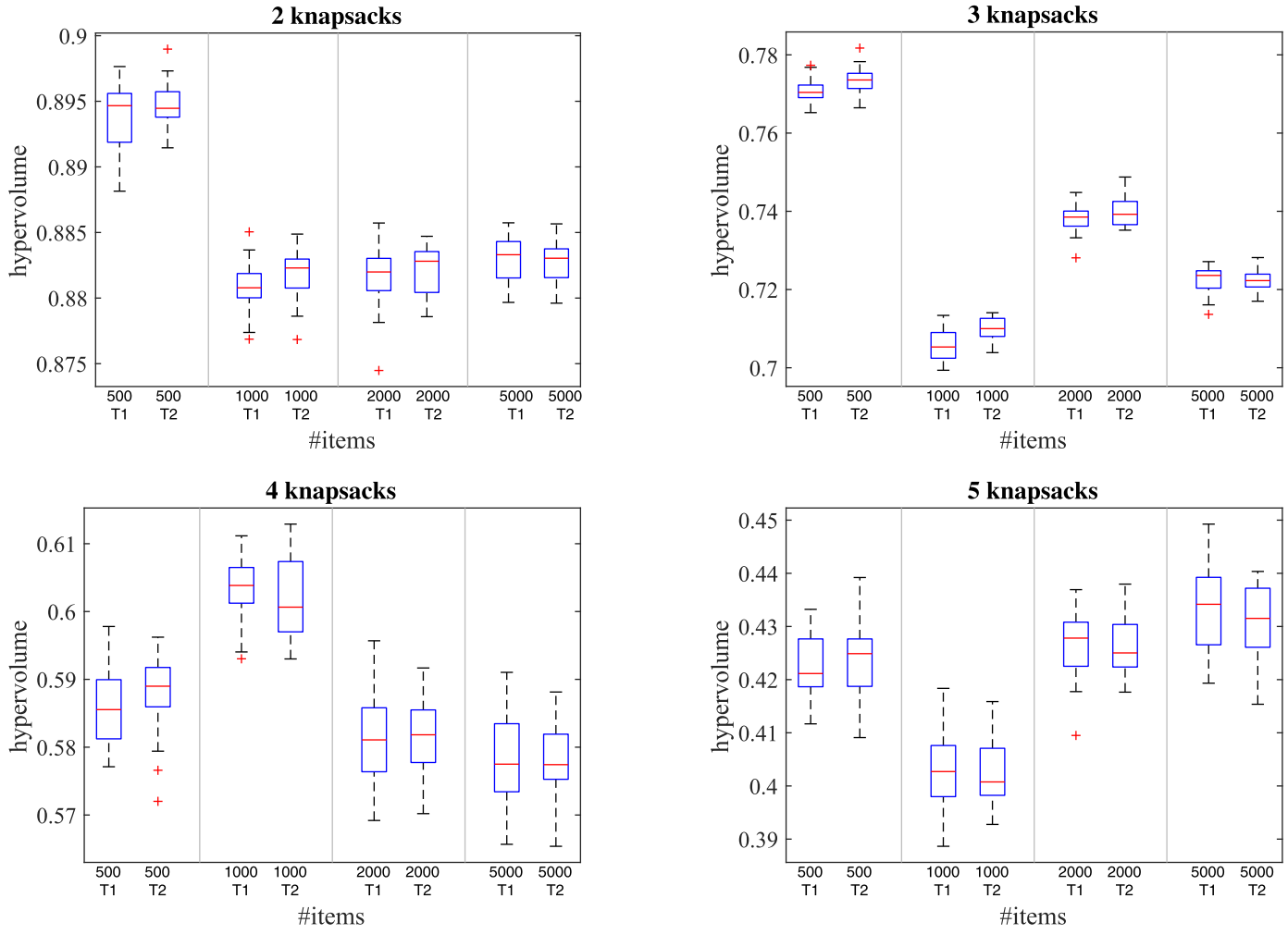

Fig. 7. Results for evolved (T1) and fixed (T2) topologies.

pair of parent chromosomes using the SBX crossover mechanism, whereas ES uses a single parent whose genes are perturbed by adding a random quantity drawn from a normal distribution with zero mean. A superior performance of ES can be explained by a self-adaptation of strategy parameters, such as the strengths of mutation for individual genes, which are encoded into the chromosome. Apparently, the evolution process benefit from discovering appropriate values of these parameters. This in turn can lead to mutations accounting for coupling between the genes. DE explores the search space using differences between chromosomes of distinct population members. DE produces an offspring by perturbing a particular individual in the population with a scaled difference of randomly selected population members. Such mechanism exhibits invariant properties that are particularly useful for dealing with nonseparable fitness landscapes (Das & Suganthan, 2011). Such kind of landscape is inherent the neural network because there is an explicit coupling between its parameters. Though, an inferior performance of DE can be explained by the need to choose appropriate values for the crossover probability *CR* and the scaling parameter *F*. In general, the obtained results stress the importance of a proper strategy for exploring the search space and suggest useful insights for their choice.

6.4. Effect of feature mapping

The proposed neuroevolution attempts to evolve neural networks so that they produce the optimal mapping between the feature and the solution space. While the solution space is determined by the MOKP at hand, the feature space is defined by the user based on the profits and weights of items in the problem. In this light, one can easily recognize that a successful solving of MOKPs is also dependent on a proper definition of the feature space. Given that, we investigate two possible variants of the function $\phi : \mathbb{R}^{2m} \longmapsto \mathbb{R}^{d}$ that for the profits and weights of each item returns its coordinates in the feature space.

The first function is defined as

$$\phi 1(\boldsymbol{p}, \boldsymbol{w}) = (p_1, \ldots, p_m, w_1, \ldots, w_m)^{\mathrm{T}}. \tag{13}$$

This is the default function that was used so far in the experiments, where the profits and weights of item are simply concatenated in a single vector. This function is simple and provides a straightforward information about each item to the network, with the number of input neurons being twice the size of the objective space.

The second function is defined as

$$\phi 2(\boldsymbol{p}, \boldsymbol{w}) = \left(\frac{p_1}{w_1}, \ldots, \frac{p_m}{w_m}\right)^{\mathrm{T}}. \tag{14}$$

This function returns a feature vector whose components are given by the ratio between profits and weights. Such function can be useful because the number of inputs and therefore the parameters in neural network is reduced by half compared with $\phi 1$.

Fig. 6 summarizes the results obtained by neuroevolution when using the feature maps defined in (13) and (14). The plots show the

median values of the hypervolume using o bar charts. These results clearly indicate a better performance of neuroevolution when $\phi 1$ is used. A possible reason for this can be that the features represented by the ratio between profits and weights are less informative than those given by profits and weights itself. However, there still remain possibilities of exploring more elaborated feature mappings that can eventually benefit the search on other MOKP instances.

6.5. Effect of network topology

In neural networks, the topology refers to the number of neurons and the way they are connected. This is an important factor because it influences the learning and determines the expressive capacity. As we use a feed forward neural network with one hidden layer, the number of neurons in the hidden layer is of major importance. This is because a small number of neurons can result in a low capacity of the neural network. However, a high number of neurons can also lead to a poor performance due to a large search space being explore by neuroevolution. Thus, there is a trade-off and we aim to address it by encoding the topology of the hidden layer into the chromosome and enabling the evolution to find the most appropriate one. This section examines the effectiveness of this approach in comparison with neuroevolution having a fixed number of neurons.

Fig. 7 shows the distributions of the hypervolume values obtained by neuroevolution with two types of topology. The box plots in this figure suggest somewhat similar performance of the two variants. Slightly better results in terms of the median and the dispersion of hypervolume values for fixed topology can be explained by the fact that this variant only explores the space defined by weights and biases and does not search the combinatorial space defined by the binary mask for hidden neurons. In contrast, neuroevolution with evolved topology explores both of these spaces, which naturally can be more difficult. An important observation is that the proposed SMS-NEMOA for solving MOKPs is not very sensitive to the choice of the network topology. Moreover, MOKP instances can be solved reasonably well without extensive experimentation to find out the most appropriate topology of neural network.

7. Conclusions

Multiobjective knapsack problem is an important combinatorial problem that frequently arises in a broad range of practical applications. Multiobjective evolutionary algorithms with binary solution representations and traditional crossovers have become a common choice for solving MOKPs. However, they can produce poor solutions with respect to convergence and diversity due to disruptive effects and a poor scalability of crossover operators. To address these issues, we suggested generating solutions to MOKPs by neural networks that classify items using the information about their profits and weights. We developed SMS-NEMOA that relies on neuroevolution to evolve neural networks and SMS-EMOA for multiobjective search.

The proposed approach significantly outperformed SMS-EMOA with traditional binary crossovers, with respect to both convergence and diversity, in the computational experiments involving MOKP instances having from 2 to 5 knapsacks with the number of items ranging from 500 to 10000. Another advantageous feature of SMS-NEMOA revealed by the experiments is its scalability. The scalability is an important requirement for practical applications because real-world MOKP instances often involve a large number of items. The observed behavior owes to the fact that the size of the search space being explored by stochastic recombination operators in the proposed neuroevolution only slightly depends on the size of the problem. This is a sharp contrast with a traditional scheme based on binary encoding and crossovers where the length of chromosome corresponds to the number of items and directly determines the search space size. We also investigated different variants of neuroevolution and gained insights that can help to improve its performance.

As future work, it would be interesting to investigate how the proposed approach generalizes. This can encompass studies with different EMOAs, variants of knapsack problems and single objective optimization scenarios. The adaptation of control parameters is another promising research direction. It can be studied how a threshold for converting neural network outputs into binary values can be adapted during the search instead of simply rounding to the nearest integer. Furthermore, an exciting research direction is to investigate how the notion of transfer learning can be exploited, as neural networks resulted from solving one MOKP can be useful when addressing other instances, possibly accelerating the search process.

Acknowledgments

This work was supported by the Portuguese “Fundação para a Ciência e Tecnologia” under grant PEst-C/CTM/LA0025/2013 (Projecto Estratégico - LA 25 - 2013-2014 - Strategic Project - LA 25 - 2013-2014).

References

Aghezzaf, B., & Naimi, M. (2009). The two-stage recombination operator and its application to the multiobjective 0/1 knapsack problem: A comparative study. *Computers and Operation Research, 36*(12), 3247–3262.
Azad, M. A. K., Rocha, A. M. A. C., & Fernandes, E. M. G. P. (2014). Improved binary artificial fish swarm algorithm for the 0-1 multidimensional knapsack problems. *Swarm and Evolutionary Computation, 14*, 66–75.
Beume, N., Naujoks, B., & Emmerich, M. (2007). SMS-EMOA: Multiobjective selection based on dominated hypervolume. *Eur. J. Oper. Res., 181*(3), 1653–1669.
Beyer, H.-G., & Schwefel, H.-P. (2002). Evolution strategies: A comprehensive introduction. *Natural Computing, 1*(1), 3–52.
Dantzig, G. B. (1957). Discrete-variable extremum problems. *Operation and Research., 5*, 266–277.
Das, S., & Suganthan, P. N. (2011). Differential evolution: A survey of the state-of-the-art. *IEEE Transaction on Evolutionary Computation, 15*(1), 4–31.
Deb, K. (2001). Multi-objective optimization using evolutionary algorithms. *Wiley-Interscience Series in Systems and Optimization*. John Wiley & Sons.
Deb, K., & Agrawal, R. B. (1995). Simulated binary crossover for continuous search space. *Complex Systems, 9*, 115–148.
Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transaction on Evolutionary Computation, 6*(2), 182–197.
Denysiuk, R., & Gaspar-Cunha, A. (2017). Weighted stress function method for multiobjective evolutionary algorithm based on decomposition. In *Proc. conf. evol. multi-criterion optim.* (pp. 176–190).
Derrac, J., García, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation, 1*(1), 3–18.
Diffie, W., & Hellman, M. (1976). New directions in cryptography. *IEEE Transaction on Information Theory, 22*(6), 644–654.
Drexl, A. (1988). A simulated annealing approach to the multiconstraint zero-one knapsack problem. *Computing, 40*(1), 1–8.
Floreano, D., Dürr, P., & Mattiussi, C. (2008). Neuroevolution: from architectures to learning. *Evolutionary Intelligence, 1*(1), 47–62.
Fontanari, J. F. (1995). A statistical analysis of the knapsack problem. *Journal of Physics A: Mathematical and General, 28*, 4751–4759.
Gavish, B., & Pirkul, H. (1985). Efficient algorithms for solving multiconstraint zero-one knapsack problems to optimality. *Math Programmin, 31*, 78–105.
Gilmore, P. C., & Gomory, R. (1966). The theory and computation of knapsack functions. *Operations Research, 14*, 1045–1075.
Glover, F., & Kochenberger, G. A. (1996). In I. H. Osman, & J. P. Kelly (Eds.), *Critical event tabu search for multidimensional knapsack problems* (pp. 407–427)). Kluwer Academic Publishers,.
Heidrich-Meisner, V., & Igel, C. (2009). Neuroevolution strategies for episodic reinforcement learning. *Journal of Algorithms, 64*, 152–168.
Ishibuchi, H., Akedo, N., & Nojima, Y. (2010a). Diversity improvement by non-geometric binary crossover in evolutionary multiobjective optimization. *IEEE Transaction on Evolutionary Computation, 14*(6), 985–998.
Ishibuchi, H., Akedo, N., & Nojima, Y. (2015). Behavior of multiobjective evolutionary algorithms on many-objective knapsack problems. *IEEE Transaction on Evolutionary Computation, 19*(2), 264–283.

Ishibuchi, H., & Kaige, S. (2003). Effects of repair procedures on the performance of EMO algorithms for multiobjective 0/1 knapsack problems. In *Proc. congr. evol. comput.: 4* (pp. 2254–2261).
Ishibuchi, H., Kaige, S., & Narukawa, K. (2005). Comparison between lamarckian and baldwinian repair on multiobjective 0/1 knapsack problems. In *Proc. conf. evol. multi-criterion optim.* (pp. 370–385).
Ishibuchi, H., Narukawa, K., Tsukamoto, N., & Nojima, Y. (2008). An empirical study on similarity-based mating for evolutionary multiobjective combinatorial optimization. *European Journal of Operational Research, 180*, 57–75.
Ishibuchi, H., Sakane, Y., Tsukamoto, N., & Nojima, Y. (2009). Effects of using two neighborhood structures on the performance of cellular evolutionary algorithms for many-objective optimization. In *Proc. congr. evol. comput.* (pp. 2508–2515).
Ishibuchi, H., Sakane, Y., Tsukamoto, N., & Nojima, Y. (2010b). Simultaneous use of different scalarizing functions in MOEA/D. In *Proc. genet. evol. comput. conf.* (pp. 519–526).
Ishibuchi, H., Tanigaki, Y., Masuda, H., & Nojima, Y. (2014). Distance-based analysis of crossover operators for many-objective knapsack problems. In *Proc. conf. parallel probl. solving nat.* (pp. 600–610).
Ishibuchi, H., Yamane, M., & Nojima, Y. (2012). Effects of discrete objective functions with different granularities on the search behavior of emo algorithms. In *Proc. genet. evol. comput. conf.* (pp. 481–488).
Jaszkiewicz, A. (2002). On the performance of multiple-objective genetic local search on the 0/1 knapsack problem - A comparative experiment. *IEEE Transaction on Evolutionary Computation, 6*(4), 402–412.
Kafafy, A., Bounekkar, A., & Bonnevay, S. (2012). Hybrid metaheuristics based on MOEA/D for 0/1 multiobjective knapsack problems: A comparative study. In *Proc. congr. evol. comput.* (pp. 1–8).
Kellerer, H., Pferschy, U., & Pisinger, D. (2004). *Knapsack problems* (1st). Berlin Heidelberg: Springer-Verlag.
Knowles, J., Thiele, L., & Zitzler, E. (2006). A tutorial on the performance assessment of stochastic multiobjective optimizers. *Technical Report*. Computer Engineering and Networks Laboratory, Swiss Federal Institute of Technology, Zurich, Switzerland.
Knowles, J. D., & Corne, D. W. (2000). A comparison of diverse approaches to memetic multiobjective combinatorial optimization. In *Proc. genet. evol. comput. conf.* (pp. 103–108).
Kong, M., Tian, P., & Kao, Y. C. (2007). A new ant colony optimization algorithm for the multidimensional knapsack problem. *Computers and Operations Research, 35*(8), 2672–2683.
Layeb, A., Lahouesna, N., & Kireche, B. (2013). A multi-objective binary cuckoo search for bi-criteria knapsack problem. *IJIEEB, 4*, 8–15.
Li, H., & Zhang, Q. (2009). Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II. *IEEE Transaction on Evolutionary Computation, 13*(2), 284–302.
Li, H., Zhang, Q., Tsang, E., & Ford, J. A. (2004). Hybrid estimation of distribution algorithm for multiobjective knapsack. In *Proc. eur. conf. evol. comput. combin. optimiz.* (pp. 145–154).
Li, K., Deb, K., Zhang, Q., & Kwong, S. (2015). An evolutionary many-objective optimization algorithm based on dominance and decomposition. *IEEE Transaction on Evolutionary Computation, 19*(5), 694–716.
Liu, H.-L., Gu, F., & Zhang, Q. (2014). Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems. *IEEE Transaction on Evolutionary Computation, 18*(3), 450–455.
Martello, S., & Toth, P. (1990). *Knapsack Problems: Algorithms and computer implementations* (Chichester, uk). Boston, USA: Wiley.
Martins, J. P., Fonseca, C. M., & Delbem, A. C. (2014). On the performance of linkage-tree genetic algorithms for the multidimensional knapsack problem. *Neurocomputing, 146*, 17–29.
Mumford, C. L. (2003). Comparing representations and recombination operators for the multi-objective 0/1 knapsack problem. In *Proc. congr. evol. comput.* (pp. 854–861).
Ohlsson, M., Peterson, C., & Soderberg, B. (1993). Neural networks for optimization problems with inequality constraints: The knapsack problem. *Neural Computation, 5*, 331–339.
Petersen, C. C. (1974). A capital budgeting heuristic algorithm using exchange operations. *AIIE Transactions, 6*, 143–150.
Pisinger, D. (1995). *Algorithms for Knapsack Problems*. University of Copenhagen Ph.D. thesis.
Rothlauf, F. (2006). *Representations for Genetic and Evolutionary Algorithms* (2nd). Heidelberg: Springer.
Sakawa, M., & Kato, K. (2003). Genetic algorithms with double strings for 0-1 programming problems. *European Journal of Operational Research, 144*(3), 581–597.
Sato, H. (2014). Inverted PBI in MOEA/D and its impact on the search performance on multi and many-objective optimization. In *Proc. genet. evol. comput. conf.* (pp. 645–652).
Sato, H., Aguirre, H., & Tanaka, K. (2013). Variable space diversity, crossover and mutation in MOEA solving many-objective knapsack problems. *Annals of Mathematics and Artificial Intelligence, 68*, 197–224.
Sato, H., Aguirre, H. E., & Tanaka, K. (2007). Local dominance and local recombination in MOEAs on 0/1 multiobjective knapsack problems. *European Journal of Operational Research, 181*(3), 1708–1723.
Shih, W. (1979). A branch and bound method for the multiconstraint zero-one knapsack problem. *Journal of Operational Research Society, 30*, 369–378.
Smith, K. A. (1999). Neural networks for combinatorial optimization: A review of more than a decade of research. *Journal of Computation, 11*(1), 15–34.
Spielberg, K. (1969). Algorithms for the simple plant location problem with some side conditions. *Operational Research, 17*, 85–111.
Stanley, K. O., D'Ambrosio, D. B., & Gauci, J. (2009). A hypercube-based indirect encoding for evolving large-scale neural networks. *Artificial Life, 15*(2), 185–212.
Stanley, K. O., & Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary Computation, 10*, 99–127.
Stanley, K. O., & Miikkulainen, R. (2003). A taxonomy for artificial embryogeny. *Artificial Life, 9*(2), 93–130.
Storn, R., & Price, K. (1997). Differential evolution: A simple and efficient heuristic for global optimization over continuous spaces. *The Journal of Global Optimization, 11*(4), 341–359.
Vaithyanathan, S., Ogmen, H., & Ignizio, J. (1994). Generalized boltzmann machines for multidimensional knapsack problems. *Intelligent Engineering System Through Artificial Neural Network, 4*, 1079–1084.
Weingartner, H. M., & Ness, D. N. (1967). Methods for the solution of the multidimensional 0/1 knapsack problem. *Operation Research, 15*, 83–103.
Y.-Y. Tan, Y.-C. J. (2013). MOEA/D with uniform design for solving multiobjective knapsack problems. *Journal of Computers, 8*(2), 302–307.
Yamamoto, A., Ohta, M., Ueda, H., Ogihara, A., & Fukunaga, K. (1995). Asymmetric neural network and its application to knapsack problem. *IEICE Transaction Fundamentals*, 300–305.
Yao, X. (1999). Evolving artificial neural networks. *Proceedings of the IEEE, 87*(9), 1423–1447.
Zhang, Q., & Li, H. (2007). MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Trans. Evol. Comput., 11*(6), 712–731.
Zhou, Y., Kuang, Z., & Wang, J. (2008). A chaotic neural network combined heuristic strategy for multidimensional knapsack problem. In *Isica* (pp. 715–722).
Zitzler, E., & Künzli, S. (2004). Indicator-based selection in multiobjective search. In *Proc. conf. parallel probl. solving nat.* (pp. 832–842).
Zitzler, E., Laumanns, M., & Thiele, L. (2001). SPEA2: Improving the strength pareto evolutionary algorithm. *Technical Report*. Zurich, Switzerland: Computer Engineering and Networks Laboratory (TIK), ETH Zurich.
Zitzler, E., & Thiele, L. (1998a). An evolutionary algorithm for multiobjective optimization: The strength pareto approach. *Technical Report*. Zürich, Switzerland: Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH).
Zitzler, E., & Thiele, L. (1998b). Multiobjective optimization using evolutionary algorithms - A comparative case study. In *Proc. conf. parallel probl. solving nat.* (pp. 292–304).
Zitzler, E., & Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE Transaction on Evolutionary Computation, 3*(4), 257–271.
Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., & Grunert da Fonseca, V. (2003). Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transaction on Evolutionary Computation, 7*(2), 117–132.
Zoua, D., Gaoa, L., Lib, S., & Wua, J. (2011). Solving 0-1 knapsack problem by a novel global harmony search algorithm. *Applied Soft Computing, 11*(2), 1556–1564.