Multidimensional Knapsack Problem: A Fitness Landscape Analysis

Jorge Tavares, *Member, IEEE*, Francisco B. Pereira, *Member, IEEE*, and Ernesto Costa

***Abstract*—Fitness landscape analysis techniques are used to better understand the influence of genetic representations and associated variation operators when solving a combinatorial optimization problem. Five representations are investigated for the multidimensional knapsack problem. Common mutation operators, such as bit-flip mutation, are employed to generate fitness landscapes. Measures such as fitness distance correlation and autocorrelation are applied to examine the landscapes associated with the tested genetic encodings. Furthermore, additional experiments are made to observe the effects of adding heuristics and local optimization to the representations. Encodings with a strong heuristic bias are more efficient, and the addition of local optimization techniques further enhances their performance.**

***Index Terms*—Fitness landscape analysis, heuristic bias, local improvement methods, representation.**

I. Introduction

EVOLUTIONARY algorithms are efficient techniques to discover good-quality solutions for difficult combinatorial optimization problems. The choice of a suitable representation plays a crucial role in the design of these biological inspired techniques and is a key issue in improving search performance [1], [2].

The main goal of this paper is to study the influence of representations on the design of efficient evolutionary algorithms for combinatorial optimization problems. The multidimensional knapsack problem (MKP) will be used as a benchmark for our study. We will use fitness landscape analysis to conduct a comprehensive investigation of the properties of different representations that are commonly adopted when evolutionary algorithms are applied to this class of problems.

Manuscript received May 22, 2007; revised November 22, 2007. The work of J. Tavares was supported by the Fundação para a Ciência e a Tecnologia, Portugal, under Grant SFRH/BD/12615/2003. This paper was recommended by Associate Editor T. Vasilakos.

J. Tavares was with the Center for Informatics and Systems of the University of Coimbra, Department of Informatics Engineering, University of Coimbra, 3030-290 Coimbra, Portugal. He is now with the French National Institute for Research in Computer Science and Control (INRIA), Research Center Lille-Nord Europe, 59650 Lille, France (email: jorge.tavares@ieee.org).

F. B. Pereira is with the Center for Informatics and Systems of the University of Coimbra, Department of Informatics Engineering, University of Coimbra, 3030-290 Coimbra, Portugal, and also with the Department of Computer Science, Instituto Superior de Engenharia de Coimbra, Polytechnic Institute of Coimbra, 3030-199 Coimbra, Portugal (e-mail: xico@dei.uc.pt).

E. Costa is with the Center for Informatics and Systems of the University of Coimbra, Department of Informatics Engineering, University of Coimbra, 3030-290 Coimbra, Portugal (e-mail: ernesto@dei.uc.pt).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCB.2007.915539

Fitness landscapes [3] illustrate the association between the search space and the fitness space. An evolutionary algorithm can be seen as navigating a landscape in order to find the highest peak. Higher points in the search space correspond to solutions with higher fitness. A representation and associated variation operators influence the search efficiency of an evolutionary algorithm. With this in mind, the choice of a representation and operators can be made based on the study of the difficulties of the corresponding fitness landscapes. Since fitness landscape analysis techniques were introduced, they have become a valuable tool in investigating how a given evolutionary algorithm works.

The MKP is a well-known combinatorial optimization problem for which several successful applications of evolutionary algorithms exist, e.g., [4]. For this problem, there is a large selection of representations. This implies that a careful analysis of representations, variation operators, and other techniques (e.g., heuristics) is essential for the design of an effective algorithm.

In this paper, we present our investigations toward a better understanding of the role of representation and heuristics applied to the MKP. The study presented in this paper continues and expands our initial work on this topic [5]. We add new results by using fitness landscape analysis techniques in mutation- and crossover-generated landscapes. We also demonstrate the point of adding heuristics and/or local improvement methods to a representation to obtain better results. To the best of our knowledge, this is the first study regarding a complete fitness landscape analysis on this problem.

This paper is structured as follows. A description and formal definition of the problem is presented in Section II. Section III presents an overview of the evolutionary techniques applied to the MKP. In Section IV, we explain the concept of fitness landscapes and describe the analysis techniques used for this paper. Experimental results and discussion are reported in Section V. Finally, in Section VI, we present some conclusions.

II. Multidimensional Knapsack Problem

The MKP is a well-known nondeterministic-polynomial-time-hard combinatorial optimization problem, with a wide range of applications, such as cargo loading, cutting stock problems, resource allocation in computer systems, and economics [6].

The problem can be described as follows. Given two sets of n items and m knapsack constraints (or resources), for each item j, a profit p_j is assigned, and for each constraint i, a consumption value r_{ij} is designated. The goal is to determine

a set of items that maximizes the total profit, not exceeding the given constraint capacities c_i. Formally, this is stated as follows:

$$\text{maximize} \quad \sum_{j=1}^{n} p_j x_j \tag{1}$$

$$\text{subject to} \quad \sum_{j=1}^{n} r_{ij} x_j \leq c_i, \qquad i = 1, \ldots, m \tag{2}$$

$$x_j \in \{0, 1\}, \qquad j = 1, \ldots, n \tag{3}$$

$$\text{with} \quad p_j > 0 \qquad r_{ij} \geq 0 \qquad c_i \geq 0. \tag{4}$$

The decision variable is the binary vector $x = (x_1, \ldots, x_n)$. Each item j is mapped to a bit. When $x_j = 1$, the corresponding item is considered to be part of the solution. The special case of $m = 1$ is generally known as the *Knapsack Problem* or the *Unidimensional Knapsack Problem*. Exact techniques and exhaustive search algorithms, such as branch-and-bound, are only of practical use in solving MKP instances of small size since they are, in general, too time-consuming (e.g., instances with 100 items or less, and depending on the constraints).

This problem is also known in the literature as the *M-Dimensional Knapsack Problem*, the *Multiconstraint Knapsack Problem*, the *Multi-Knapsack Problem*, or the *Multiple Knapsack Problem*. Some authors also include in their description the term *zero–one*, e.g., the *Multidimensional zero–one knapsack problem*. Using alternative names for the same problem is potentially confusing, but since, historically, the designation MKP has been the most widely used [4], we adopt this same name in the following discussion.

The MKP has been widely studied in the past few years, and many theoretical and empirical studies exist for a different number of knapsack problem variants. For a comprehensive review of these techniques, including exact methods and heuristics, consult [4], [6], or [7]. In the next section, we will present and discuss the most common evolutionary approaches to the problem.

III. Evolutionary Approaches

Evolutionary algorithms have been widely applied to the MKP and shown to be effective for searching and finding good-quality solutions (consult, e.g., [4]). Regardless of that success, designing an efficient evolutionary algorithm for the MKP is a difficult task, where the issue of choosing an appropriate constraint-handling technique is important. As shown in [8], the success of an evolutionary algorithm for the MKP is strongly dependent on the ability of the algorithm to restrict, or focus, the search to the boundary of a feasible region.

As such, the most successful evolutionary algorithms are based on repairing and local optimization techniques or heuristic decoders, such as, for example, the approaches developed in [4] and [9]. The generality of the proposed algorithms found in the literature [4], [10]–[13] can be divided in two groups according to the adopted representation and the associated variation operators. Evolutionary algorithms may use a direct representation, or alternatively, they may adopt indirect encodings. In this case, a decoder that translates the chromosome into the actual solution is needed.

In the following sections, we briefly describe the most common representations for this problem and a simple description of some significant works regarding evolutionary algorithms for this problem.

A. Binary Representation

With this encoding, a solution is represented by a characteristic bit vector, where each bit is mapped to an item. A bit set to 1 indicates that the corresponding item is packed into the knapsack. For the MKP, two options are commonly used to deal with unfeasible solutions: penalty-based fitness functions (consult [14] for a comprehensive study on fitness functions for the MKP) or repair operators. In [4], a repair mechanism that iteratively removes items until all constraints are satisfied is proposed. In addition, each solution is improved by local optimization. Both methods are guided by a heuristic that orders the items according to the ratio of profit and resource consumption. This approach was later improved by Raidl [9] by using the heuristic associated with the weight-coding representation, which will be described later. For a binary representation, classical crossover and mutation operators can be used, such as n-point crossover or uniform crossover, and bit-flip mutation.

B. Ordinal Representation

Here, a chromosome is a vector $v = (v_1, \ldots, v_n)$, where each position v_k belongs to the set $\{1, \ldots, n - k + 1\}$ for $k \in \{1, \ldots, n\}$. The vector is mapped to a permutation π of the items. This permutation is built in the following manner: An ordered list $L = (L_1, \ldots, L_n)$ is created and initialized with all the items. Next, vector v is traversed from the first to the last position. Each v_k specifies a position in L. The referenced element L_{v_k} is removed from L and inserted in permutation π, where it represents element π_k. To give an example, assume that $v = (1, 2, 3, 2, 1)$ and the initial ordered list $L = (1, 2, 3, 4, 5)$. Vector v is interpreted by successively removing elements 1, 3, 5, 4, and 2 from L. This process generates permutation $\pi = (1, 3, 5, 4, 2)$. To decode the permutation into a feasible MKP solution, a *first-fit* heuristic is applied. The heuristic works by building a feasible solution traversing all variables in the order given by permutation π. An item π_j is inserted into the solution if it does not violate any constraint. To be more precise, we start with an empty solution $x = (0, \ldots, 0)$. For each item in the order given by permutation π, the corresponding decision variable x_{π_j}, with $j = 1, \ldots, n$, is altered from 0 to 1 if the insertion of item π_j does not violate any constraint.

This representation has the advantage of enabling the use of standard genetic operators. Classical crossover operators, e.g., uniform crossover, can be used. The mutation operator randomly chooses a position $k \in \{1, \ldots, n\}$ and then picks a new value v_k using a uniform distribution, from the set $\{1, \ldots, n - k + 1\}$. Evolutionary algorithms for the MKP using this representation can be found in [10] and [15].

These studies report poor results in comparison with other representations.

C. Permutation Representation

Permutations are typical representations for scheduling and routing problems. Additionally, they have been widely applied to the MKP [9], [11]–[13]. The representation consists of a permutation of all items $\pi : \{1, \ldots, n\} \rightarrow \{1, \ldots, n\}$ denoted by $\pi = (\pi_1, \ldots, \pi_n)$. Decoding it into a feasible solution is done by a *first-fit* heuristic, in the same manner as in the ordinal representation. The representation needs genetic operators that can preserve the permutations, such as uniform-order-based crossover [15] or partially matched crossover, and swap mutation [16]. Permutation-based evolutionary algorithms have achieved good results, as reported by Hinterding [11] and Gottlieb [13].

D. Random-Key Representation

Proposed by Bean [17], the random-key representation is an alternative approach in encoding permutations without the need for specific operators. A random key is a vector of real values $w = (w_1, \ldots, w_n)$, where each gene represents an item j, which is assigned by a weight $w_j \in [0, 1]$. The decoder works by sorting the real-valued vector, yielding a permutation π with the associated weight indexes. Once again, the attained permutation is decoded by means of a *first-fit* heuristic. This representation allows the use of the classical crossover and mutation operators. In the case of mutation, the operator usually performs positional mutation, i.e., it randomly draws values from the interval [0, 1] according to a distribution, replacing a value at a random position in the chromosome. Evolutionary approaches based on random keys for MKP can be found in [11] and [18]. The reported results show that evolutionary approaches based on this representation can achieve good results.

E. Weight-Coding Representation

The weight-coding representation, a general technique successfully applied to a variety of combinatorial optimization problems, is the most successful decoder-based technique for the MKP. It was first used by Cotta and Troya [19] and later improved by Raidl [20]. In this representation, a chromosome consists of a real-valued vector of weights $w = (w_1 \ldots, w_n)$, where each item j of the MKP is associated with a weight $w_j \in [0, 1]$. The decoding process of the genotype to the phenotype is made in two steps: The first step consists of transforming the original problem P into a modified problem P' by multiplying the original items profits by the associated weight, i.e., biasing the original problem. The last step requires the use of a fast heuristic to find a solution to P' and to evaluate it according to the original problem.

Several decoding heuristics and techniques for biasing the original problem have been studied [20]. The heuristics that often work best are based on surrogate and Lagrangian relaxation techniques. The decoding heuristic using the surrogate relaxation method is preferred due to its lower computational requirements. The original problem is simplified by transforming all m constraints into a single constraint, i.e.,

$$\sum_{j=1}^{n} \left(\sum_{i=1}^{m} a_i r_{ij} \right) x_j \leq \sum_{i=1}^{m} a_i c_i \tag{5}$$

where a_i, $i = 1, \ldots, m$, is the surrogate multiplier for the ith constraint. To derive the surrogate multipliers, one of the simplest methods is to solve the linear programming (LP) relaxation of the original problem (i.e., decision variables x_i can take any value $\in [0, 1]$) and to use the values of the *dual variables* as the surrogate multipliers. To obtain a heuristic solution to the MKP, the *profit/pseudoresource consumption ratios* u_j are calculated as

$$u_j = \frac{p'_j}{\sum_{i=1}^{m} a_i r_{ij}} \tag{6}$$

where $p'_j = p_j \times w_j$. A low pseudoutility ratio heuristically indicates that an item has a lower profit with a higher resource consumption, whereas a high ratio reflects a more efficient item. According to the u_j values, we sort the items in decreasing order, adding them to the solution one at a time if none of the constraints are violated. This process is similar to the *first-fit* heuristic previously described in the decoding steps for encodings where permutations are involved. The surrogate multipliers for the original problem are only calculated once as a preprocessing step performed in the beginning. This is to ensure low computation costs, and as a result of this step, the decoding process begins by determining the u_j values.

This representation allows the use of standard crossover and mutation variation operators. Like the random-key representation, positional mutation is used. For details of this method, please refer to [20]. This approach has attained good results when applied to the MKP.

F. Related Work

Although many different evolutionary algorithms have been applied to the MKP, there are just a few studies that aim to perform a comprehensive analysis of the behavior of these techniques.

The most significant work on this subject was done by Raidl and Gottlieb [21]. In this study, five representations (four indirect decoder-based representations and one direct representation) are examined according to important aspects of an evolutionary algorithm, such as initialization and variation operators. The study comprises a general performance comparison of the five encodings, as well as a comprehensive empirical investigation of different characteristics of an evolutionary algorithm. In this case, three properties, i.e., locality, heritability, and heuristic bias, are considered. A set of static measures is used to gain insights about the behavior of the evolutionary algorithm when using the tested representations.

In a previous work [5], we used the fitness landscape framework to initiate a study with representations for the MKP. The analysis also contained some repair operators, local optimization, and heuristics. The focus of that investigation was on

fitness landscapes generated by mutation operators. The analysis of fitness distance correlation and autocorrelation was useful in gaining some insights about the differences of performance achieved by different representations. The study shows that the choice of an encoding without a strong heuristic bias can create some difficulties for an evolutionary algorithm designed for the MKP. In this paper, we continue and expand the investigation previously published by extending it and adding a more clear and deep overall analysis.

IV. Fitness Landscapes

The concept of *fitness landscape*, which was introduced by Wright [3] to demonstrate the dynamics of biological evolutionary optimization, has been useful for the analysis and understanding of evolutionary algorithm's behavior. In addition, the study of fitness landscapes can be of value in designing an evolutionary algorithm since it can help predict its performance. Usually, evolutionary search can be represented by three spaces: 1) the *search space*; 2) the *phenotype space*; and 3) the *fitness space*. The fitness space reflects the solution quality, whereas the search space is made of the candidate solutions. The set of all possible genotypes is denominated *genotype space*, and it is equivalent to the search space. This equivalence is possible, because variation operators work in the genotype space and an evolutionary algorithm searches for genotypes that decode into phenotypes with high fitness. Fitness landscapes describe the relation between the *search space* and the *fitness space*. Regarding the *search space* as a landscape, an evolutionary algorithm can be seen as navigating through it in order to find its highest peak. The height of a point in the *search space* (the genotype) reflects the fitness of the decoded solution (the phenotype) associated with that point. Since the representation and operators define how an evolutionary algorithm can perform its search, the choice of a representation and operators for a given problem can be based on the study of the difficulties of the corresponding landscape.

A fitness landscape is a tuple composed of a set of points (solutions) X; a fitness function f, which assigns a numeric value to each solution; and a neighborhood $\mathcal{N}_k$ defined over set X, which is given by a distance metric of size k. The following tuple can represent the fitness landscape:

$$\mathcal{L} = (X, f, \mathcal{N}_k) \tag{7}$$

where

$$f : X \to \mathcal{R} \tag{8}$$

$$\mathcal{N}_k(x) = \{y \in X : d(x, y) \leq k\} \tag{9}$$

with the distance metric defined as the minimum number k of applications of an elementary operator m to transform one solution in to another.

A. Measures for Landscape Analysis

There are several properties that define the structure of fitness landscapes, such as the distribution of the fitness function, the number and distribution of local optima, the structure of the basins of attraction, the presence and structure of neutral networks, and landscape ruggedness. These characteristics are well known and have been studied in the evolutionary computation community, and several methods have been proposed to measure some of these properties [22]–[24].

1) Fitness Distance Correlation: One way of measuring problem difficulty is determining how close is the relation between fitness value and distance to the nearest optimum, in the search space. The fitness distance correlation, which is called coefficient ϱ, can be estimated by

$$\varrho(f, d) \approx \frac{1}{\sigma_f \sigma_d m} \sum_{i=1}^{m} (f_i - \overline{f})(d_i - \overline{d}) \tag{10}$$

with a given set of points x_i of size m (the random walk length), $f_i = f(x_i)$ being the fitness value, and $d_i = d_{\text{opt}}(x_i)$ being the minimum distance to a global optimum solution. The $\overline{f}$ and σ_f are the mean and standard deviation, respectively. The search should be easy, for selection-based algorithms, when fitness increases as the distance to the optimum decreases. This indicates the existence of a *path* via solutions with increasing fitness values. A value of -1.0 for ϱ shows that fitness and distance to the optimum are perfectly related; thus, the search is easy, indicating a strong correlation. A value of $\varrho = 1.0$ indicates the opposite.

2) Autocorrelation Function: The structure of a fitness landscape can be examined by measuring the degree of correlation between points on the landscape. The degree of correlation depends on the difference between the fitness values of the points. Smoother landscapes are highly correlated, making the search for a search algorithm easier. This is the result of similar fitness values. If the difference of fitness values is higher, the landscape is less correlated, which implies a rugged landscape, thus making the search more difficult. To estimate $\rho(d)$, we perform a random walk. In this case, random walk $\langle f(x_t) \rangle$ defines the correlation of two points s steps away in an m-long random walk, i.e.,

$$\rho_{rw}(s) \approx \frac{1}{\sigma_f^2 (m - s)} \sum_{t=1}^{m-s} \left(f(x_t) - \overline{f} \right) \left(f(x_{t+s}) - \overline{f} \right). \tag{11}$$

3) Correlation Length: This measure directly represents the ruggedness of a fitness landscape: The higher the value of correlation length l, the smoother the landscape; the lower the value of l, the more rugged the landscape. The correlation length is defined as

$$l = -\frac{1}{\ln(|\rho(1)|)} \tag{12}$$

for $\rho(1) \neq 0$. It is useful to normalize it with the diameter of the landscape. The closer the normalized correlation length is to 1, the higher the correlation is. If $\xi = 0$ or close to 0, there is no correlation.

4) Crossover Measures: Studying the behavior of mutation-based algorithms using the aforementioned methodology is simple and well suited. The reason is that mutation operators

are good in generating random walks for correlation analysis. These operators can be seen producing a time series of fitness values for a succession of solutions [25]. Normally, a mutation operator is more appropriate to a certain landscape if the correlation is higher. However, for crossover-generated landscapes, this is a completely different issue. In this paper, we followed an approach that is similar to the one defined by Hordijk and Manderick [26]. A time series is simply produced by repeatedly applying crossover to two candidate solutions. Given the parents and the offspring, we measure the fitness and distance to produce the landscape. The previous fitness distance correlation measure and the correlation are updated for crossover analysis. For both measures, we take into consideration the average fitness of both parents f_p, the average fitness of both offspring f_o, and the average distance of the children to the known optimum solution d_{opt}. The measures for analysis are redefined in the following way. The fitness distance correlation for crossover ϱ_{cx} can be estimated by

$$\varrho_{cx}(f_o, d_{\text{opt}}) \approx \frac{1}{\sigma_{f_o}\sigma_{d_{\text{opt}}} m} \sum_{i=1}^{m} (f_{o_i} - \overline{f_o})(d_{o_i} - \overline{d_{\text{opt}}}) \quad (13)$$

with a given pair of points x_i of size m (the random walk length). As previously defined for mutation landscapes, a value of -1.0 for ϱ_{cx} shows that fitness and distance to the optimum are perfectly related, indicating an easy search, whereas a value of $\varrho_{cx} = 1.0$ signals the opposite.

As in [22], the parent–offspring correlation is given by

$$\rho_{cx} = \frac{\text{cov}(f_p, f_o)}{\sigma_{f_p}\sigma_{f_o}} \quad (14)$$

where $\text{cov}(f_p, f_o)$ is the covariance of two variables, which are, in this particular case, the average fitness of the parents and the offspring.

B. Problem Difficulty and Limitations

The measures for fitness landscapes previously described can be used to analyze evolutionary algorithms, but these must be related to the concept of *problem difficulty*. Several attempts have been made to find problem characteristics that make a problem hard for an evolutionary algorithm and, as such, also a measure for problem difficulty. Knowing these features is important since not only will it enable the development of new ways to measure how well an algorithm performs but it also allows to better adequate current measures to study and analyze evolutionary algorithms.

Although fitness landscape analysis can be useful in performance prediction and analysis for evolutionary algorithms [25], it has a limitation. For problems with unknown global optima, the measure is no longer directly applicable. It can be approximated, but it may lead to incorrect predictions, as shown by Reeves [24] and Altenberg [27]. Regardless of the limitations found, fitness landscape analysis is a valuable tool to the study and design of evolutionary algorithms, as several researchers have illustrated [22], [25], [28].

V. MKP Landscape Analysis

We consider five representations: 1) binary; 2) ordinal; 3) permutation; 4) random key; and 5) weight-coding. For each representation, we will study landscapes generated by both types of variation operators: mutation and crossover. Standard mutation operators are used to create landscapes for each one of the representations. These are bit-flip mutation, integer flip mutation, swap, and uniform flip mutation. As for landscapes generated by crossover, standard one-point and uniform crossover operators are applied. The only exception is related to the permutation encoding. In this case, both one-point and uniform crossovers are altered to allow the manipulation of permutations [13]. All representations cover the feasible region of the search space, with the exception of the direct encoding. Therefore, we use a penalty function, as recommended by Gottlieb [14], since it can better guide the search to the feasible region.

In this paper, we consider the binary representation in different forms. First, we will be concerned with the simple binary encoding with penalty-based fitness function. Then, we will add the *profit/pseudoresource consumption ratios* to the fitness function to obtain a better solution. Finally, we will add repair and local optimization methods. This is accomplished to better examine the effect of heuristics and local search on the encoding. It is important to observe the effect of adding these methods to a representation. We begin our analysis with standard and simple encodings that do not use heuristics particularly biased toward fitter phenotypes.

A. Experimental Setup

For our analysis, we selected problem instances from two different MKP test suites available from the *OR-Library*[1] as well as another benchmark suite provided by Glover and Kochenberg.[2] We performed a comprehensive set of experiments with instances from both suites. We present results from some selected instances. Results obtained with other examples follow the same trend. From the first suite, we will analyze the results from instances containing 28 and 50 items, with ten and five constraints, respectively (named P01 and P02). From the second data set, we selected instances with 100 items and 5 constraints, and 25% of tightness (α). This means that $c_i = \alpha \sum_{j=1}^{n} r_{ij}$ holds for all $i \in I$. These instances are designated as CB01 and CB02. From the last benchmark suite, we chose the instance with 100 items and 15 constraints (labeled GK01).

A fitness distance analysis requires that the global optima are known (or a very near-global optimum solution). For these instances, we solved them to optimality by running a mixed integer programming solver from the GNU Linear Programming Kit.[3] We only consider the existence of one global optimum. Additionally, the distance between the known optimal solution and the candidate solutions in our random walks is calculated at the phenotype level. This simply means that, when using indirect encodings, all solutions contained in a random walk are converted to its phenotype. The distance is given by

[1] http://people.brunel.ac.uk/mastjjb/jeb/info.html
[2] http://hces.bus.olemiss.edu/tools.html
[3] http://www.gnu.org/software/glpk/glpk.html

TABLE I
SUMMARY RESULTS FOR MUTATION LANDSCAPES

Representation	Instance			Measures			
	Name	n	m	$\overline{d_{opt}}$	ϱ	l	ξ
BR	P01	28	10	14.03 (0.36)	0.07	7.12	0.25
OR	P01	28	10	12.02 (0.35)	-0.51	4.61	0.16
PR	P01	28	10	12.01 (0.35)	-0.51	7.64	0.27
RK	P01	28	10	12.03 (0.36)	-0.53	10.97	0.39
WC	P01	28	10	**7.53** (0.34)	**-0.58**	**14.51**	**0.52**
BR	P02	50	5	25.01 (0.49)	0.09	16.18	0.32
OR	P02	50	5	20.67 (0.57)	**-0.45**	6.23	0.12
PR	P02	50	5	20.64 (0.53)	-0.41	11.72	0.23
RK	P02	50	5	20.72 (0.54)	-0.28	17.05	0.34
WC	P02	50	5	**17.48** (0.46)	-0.37	**19.43**	**0.39**
BR	CB01	100	5	49.95 (0.76)	-0.05	**35.74**	**0.36**
OR	CB01	100	5	49.70 (0.71)	-0.57	4.09	0.04
PR	CB01	100	5	49.64 (0.71)	-0.53	7.04	0.07
RK	CB01	100	5	49.63 (0.72)	-0.56	8.12	0.08
WC	CB01	100	5	**40.85** (0.69)	**-0.62**	10.85	0.11
BR	CB02	100	5	50.00 (0.64)	-0.02	**36.65**	**0.37**
OR	CB02	100	5	49.78 (0.71)	**-0.60**	4.78	0.05
PR	CB02	100	5	49.81 (0.70)	-0.54	6.89	0.07
RK	CB02	100	5	49.64 (0.67)	-0.53	9.22	0.09
WC	CB02	100	5	**40.68** (0.74)	-0.59	10.78	0.11
BR	GK01	100	15	49.95 (0.75)	0.06	**27.32**	**0.27**
OR	GK01	100	15	49.99 (0.70)	-0.15	3.95	0.04
PR	GK01	100	15	49.99 (0.77)	-0.12	8.37	0.08
RK	GK01	100	15	49.91 (0.77)	**-0.19**	8.98	0.09
WC	GK01	100	15	**49.41** (0.74)	-0.08	9.33	0.09

the Hamming distance between the solution and the known optimum solution. For each random walk, we performed 50 runs with 100 000 steps.

B. Mutation Fitness Landscapes

In Table I, we present the results of the applied measures, regarding fitness distance correlation and autocorrelation for mutation landscapes for all five MKP instances. The first column indicates the representation used (BR, binary representation; OR, ordinal representation; PR, permutation representation; RK, random-key representation; WC, weight-coding representation), n indicates the number of items, and m indicates the number of resources. The average of the minimum distance between the optimum solution and the best found individual per random walk is denoted as $\overline{d_{\mathrm{opt}}}$, with the standard deviation in brackets. Table I also shows fitness distance correlation ϱ, correlation length l, and normalized correlation length ξ. The values in bold give the best representation for a given problem instance.

1) Distance to Optimum: The data presented in Table I show that in terms of the average distances between the best found individual and the optimal solution, binary representation has the worst behavior, whereas weight-coding has the best results. In terms of percentage, the distance from the optimum for binary encoding is always about 50% (measured in relation to the maximum distance, i.e., n), whereas that for the ordinal, permutation, and random-key representations is about 42% and that for weight-coding is in the range of 26%–35%. This pattern is not found for the larger instances. For all the instances with $n = 100$, the average distance to the optimum solution is very similar for all tested representations, with values of about 50%, except for weight-coding, which shows slightly better values on instances CB01 and CB02 but similar values as the other encodings for instance GK01. Binary encoding has this behavior since it is the only encoding that cannot produce solutions within the boundary of feasible solutions. It is interesting to notice that, as n increases, the behavior gap between the different representations decreases. This effect might be explained, to a certain degree, with the tightness of the different instances. The smaller instances have a higher tightness value than the larger ones. This could be an indication that the heuristic bias, which is not present in the binary encoding, is important for tighter instances. On the other hand, it can also be due to the increasing size of the search space when it comes to larger instances.

2) Fitness Distance Correlation: When looking at the fitness distance correlation coefficient, the observed pattern is different. One might expect that the weight-coding representation with the uniform flip operator would perform better in comparison to the other encodings, simply because it has a stronger heuristic bias on its decoder. From column ϱ, it is clear that this does not happen. The weight-coding scheme only has the best coefficient value for instances P01 and CB01 and a close second best for instance CB02. For the other two instances P02 and GK01, the fitness distance correlation for weight coding is worse than that for ordinal and random-key encodings, respectively. The only clear pattern presented for this column is the poor behavior of the binary encoding. In contrast to the previous measurement, the fitness distance correlation does not improve as n increases. With values near 0.0, ranging from 0.09 to -0.05, the order of magnitude cannot be compared with the other representations. In general, the other four representations have similar values around -0.50, with the exception of instance GK01. For the instance with more constraints, the ϱ values are all above -0.20, indicating that, for this particular instance, all representations had difficulties to reach better solutions.

From this table, it is possible to distinguish between the decoder-based encodings and the direct encoding, which is not an unexpected result. Differences between the three decoder-based encodings with the first-fit heuristic and the decoder with the profit/pseudoresource consumption ratios are less visible. There is an indication that the weight-coding representation performs slightly better than the others, but that might also be dependent on the problem instance.

3) Autocorrelation: Fitness distance correlation has also been examined in association with the autocorrelation measures. To determine the correlation length, we performed a series of random walks to calculate the autocorrelation function with a distance of 1. The neighborhood operator used was based on the Hamming distance between two bit strings (for all representations, after decoding them into a bit string). The results of the autocorrelation analysis are also presented in Table I. A brief overview of the results reveals an interesting pattern. For instances P01 and P02, weight-coding representation achieved the highest correlation value, whereas, for instances CB01, CB02, and GK01, binary representation attained the highest values. Ordinal encoding shows the lowest autocorrelation values for all problem instances. Another fact is the correlation for all decoder-based representations on instances with 100 items.

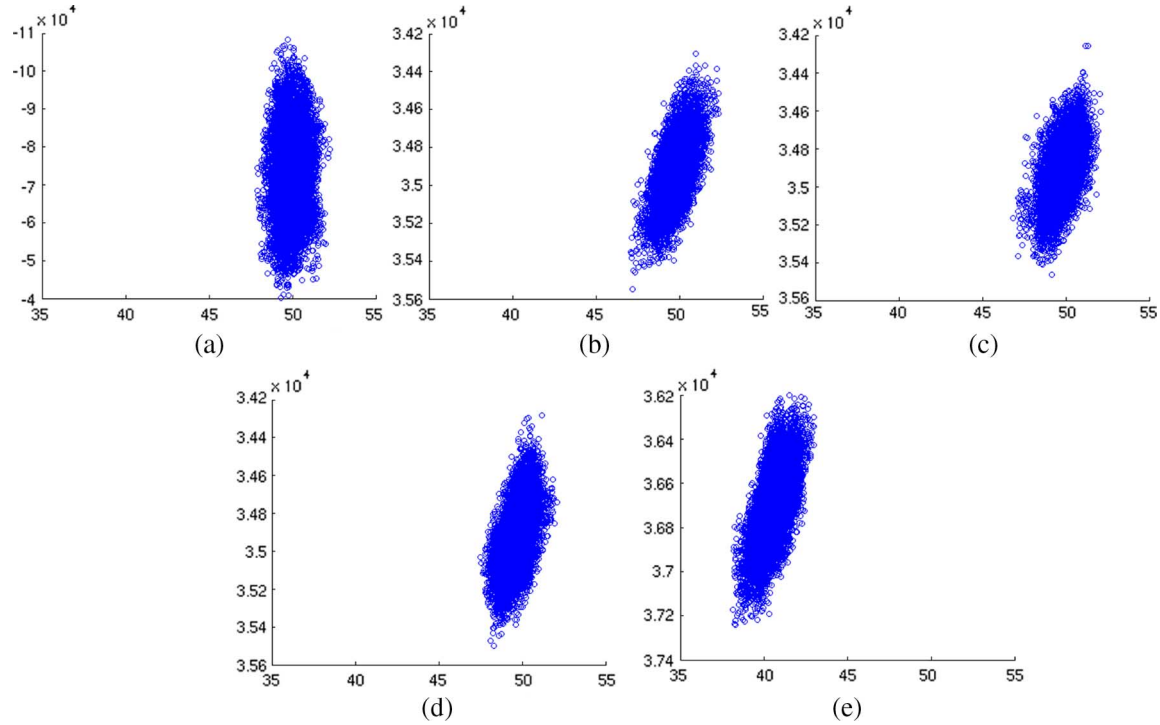


Fig. 1. Fitness distance plots for (a) binary, (b) ordinal, (c) permutation, (d) random-key, and (e) weight-coding representations on the CB02 instance. The five graphs plot the *distance to the optimum* (on the x-axis) versus the *fitness value* (on the y-axis).

Here, the values are similar and lower, when compared to the direct encoding. In fact, binary representation shows consistent values across the instances (between 0.25 and 0.37), which does not happen for the indirect encodings.

How can these differences be explained? Part of the answer relies on the type of mutation operator. The lower correlation for the ordinal representation is consistent with previous studies [21]. Locality indicates that small variations in the genotype space, which are usually originated by mutation, imply small variations in the phenotype space. A strong locality allows a search algorithm to efficiently explore the neighborhood of the current solutions, whereas a weak locality prevents evolutionary search from a meaningful exploration of the phenotype space, because small variations often cause strong phenotypic changes. The weak locality for this representation means that a single change in one of its genes can cause a major effect on the decoded solution, thus the resulting lower correlation, which implies a more difficult search. This does not happen for the other representations with the same emphasis. The flip mutation operator also changes a single gene, but the effect on the phenotype is not as dramatic as that for the ordinal representation (with the exception of permutations). This effect can explain why ordinal representation has the worst correlation. Furthermore, binary representation achieves higher correlations since the occurrence of mutation can change its genotype without causing a major disruption on its phenotype, i.e., this representation has a higher locality. This effect is more evident on the larger instances. In addition, for indirect representations, many genotypes are mapped to the same phenotype, i.e., a mutation may not change the phenotype at all. In this case, the weight-coding and binary representations are shown to have better correlated landscapes than the other representations.

4) Fitness Distance Plots: Moreover, it is also important to observe the fitness distance plot of several representations. These plots present information about the distribution of the random walk points, where the fitness is plotted against the minimum distance to an optimum. The fitness distance plot provides additional insight since it contains more information. Fig. 1 shows the fitness distance plots for binary, ordinal, permutation, random-key, and weight-coding representations for the CB02 instance (which is representative for the analyzed MKP instances). From the five plots, the distribution of the candidate solutions induced by the encodings can be observed. The first conclusion that can be drawn from these plots is that none of the representations produce highly correlated landscapes. The landscapes do not have an ideal distribution of points since the optimum solution cannot be found by *jumping* from one solution to a better one with successively decreasing the jump distance.

Although the plots show that fitness and distance are correlated with similar distribution shapes, not all representations present the same configuration. Binary representation does not show a concentration that is closer to the optimum as the others. The y-axis is negative and has much higher absolute values. This can be explained by the fact that much more unfeasible solutions are generated, independently of the penalty function. Nevertheless, in this case, the encoding allows the presence of unfeasible solutions, which cannot be ignored. The weight-coding representation is closer to the optimum. In fact, the fitness distance plot reveals that local minima are found only in a fraction of the search space with smaller distance to the optimum. All other encodings have a larger distance to the optimum. Ordinal, permutation, and random-key encodings show identical plots.

TABLE II
SUMMARY RESULTS FOR CROSSOVER LANDSCAPES

Representation	Instance			One point Crossover			Uniform Crossover		
	Name	n	m	$\overline{d_{cx_{opt}}}$	ϱ_{cx}	ρ_{cx}	$\overline{d_{cx_{opt}}}$	ϱ_{cx}	ρ_{cx}
BR	P01	28	10	13.23 (0.36)	-0.04	0.72	13.46 (0.37)	-0.03	0.54
OR	P01	28	10	10.46 (0.35)	-0.47	0.83	10.87 (0.35)	-0.48	0.68
PR	P01	28	10	12.12 (0.36)	-0.51	**0.90**	10.96 (0.35)	-0.50	0.81
RK	P01	28	10	10.55 (0.35)	-0.51	0.87	10.99 (0.35)	-0.49	0.82
WC	P01	28	10	**5.66** (0.30)	**-0.57**	**0.90**	**6.05** (0.31)	**-0.55**	**0.86**
BR	P02	50	5	24.53 (0.51)	0.01	0.78	24.65 (0.50)	0.01	0.65
OR	P02	50	5	19.20 (0.51)	-0.26	0.81	19.51 (0.52)	-0.29	0.62
PR	P02	50	5	20.87 (0.53)	-0.37	**0.90**	19.58 (0.53)	-0.32	0.78
RK	P02	50	5	19.19 (0.51)	-0.28	0.79	19.58 (0.53)	-0.34	0.78
WC	P02	50	5	**16.31** (0.45)	**-0.40**	0.82	**16.48** (0.44)	**-0.39**	**0.80**
BR	CB01	100	5	49.64 (0.72)	-0.02	0.80	49.74 (0.71)	-0.01	**0.75**
OR	CB01	100	5	46.37 (0.66)	**-0.58**	0.79	47.66 (0.68)	-0.59	0.55
PR	CB01	100	5	49.59 (0.71)	-0.56	**0.91**	47.58 (0.69)	**-0.61**	0.72
RK	CB01	100	5	46.09 (0.66)	**-0.58**	0.74	47.55 (0.69)	**-0.61**	0.72
WC	CB01	100	5	**37.52** (0.64)	-0.56	0.75	**37.87** (0.64)	-0.55	0.71
BR	CB02	100	5	49.62 (0.71)	-0.03	0.82	49.72 (0.71)	-0.03	**0.75**
OR	CB02	100	5	46.47 (0.66)	-0.58	0.81	47.73 (0.67)	-0.57	0.57
PR	CB02	100	5	49.87 (0.71)	-0.56	**0.91**	47.76 (0.69)	**-0.62**	0.73
RK	CB02	100	5	46.25 (0.67)	**-0.59**	0.76	47.74 (0.69)	-0.61	0.73
WC	CB02	100	5	**36.96** (0.64)	-0.57	0.76	**37.37** (0.66)	-0.57	0.73
BR	GK01	100	15	49.89 (0.71)	-0.02	0.78	49.91 (0.71)	0.01	**0.72**
OR	GK01	100	15	49.33 (0.73)	-0.09	0.78	49.48 (0.72)	-0.08	0.52
PR	GK01	100	15	49.86 (0.71)	-0.09	**0.90**	49.53 (0.72)	**-0.10**	0.67
RK	GK01	100	15	49.35 (0.72)	-0.09	0.71	49.39 (0.71)	-0.08	0.65
WC	GK01	100	15	**48.92** (0.71)	**-0.10**	0.71	**48.95** (0.71)	-0.08	0.65

C. Crossover Fitness Landscapes

In Table II, we present the results regarding crossover analysis. Table II gathers data from one-point crossover and uniform crossover. Uniform crossover uses probability $p = 0.5$ for directly transferring the gene of the parent to the offspring, which is a commonly used value. As before, the first column indicates the representation used, the second column indicates the tested instance, and the last two columns provide the data for the crossover operators. For each operator, we show the average of the minimum distance between the optimum solution and the average offspring found in the random walk $\overline{d_{cx_{\mathrm{opt}}}}$, with the standard deviation in brackets. The values for the fitness distance correlation are given by ϱ_{cx}, and the parent–offspring correlation is denoted as ρ_{cx}.

1) Fitness Distance Correlation: A first inspection of the results allows us to observe some important aspects. By looking at the best values, which are marked in bold, it is clear that there is not one outstanding representation. The exception is the case when we look at the distance between the offspring and the optimum solution. In that situation, in every single instance, weight coding is better than all the other encodings. The differences between the other representations are small, particularly for the case of the ordinal, permutation, and random-key representations. Furthermore, it seems that the type of instance has an effect on the representation behavior. Considering the fitness distance correlation, the instances from the first data set P01 and P02 are also dominated by the weight-coding representation. When looking at the remaining instances, weight-coding is not able to perform better than other encodings. In fact, encodings that make use of a *first-fit* heuristic in their decoding process from genotype to phenotype seem to be more suitable. This is particularly visible in the random-key representation, as it has the best fitness distance correlation values for instances CB01 and CB02.

In addition, the results found in Table II reveal that binary representation has inferior values when compared to all other representations, regardless of the problem instance. Above all, the fitness distance correlation values for binary representation are considerably lower. They are near 0.0, whereas, for other representations, the values are, on average, close to -0.5 (with the clear exception of GK01).

Fitness distance plots are similar to plots of mutation landscapes in terms of distribution shape. In spite of this, minor differences between plots of weight-coding exist: The crossover plots are closer to the optimum, which can be explained by the stronger heuristic bias in this encoding.

2) Parent–Offspring Correlation: As for the parent–offspring correlation, the correlation between parents and their offspring is similar for all representations, attaining some high values. Even for the binary encoding, the correlation values are good—a fact already observed in the mutation analysis. Only ordinal encoding presents slightly inferior values when compared to the other representations. As a final observation, neither crossover operator demonstrates a superior behavior. How can these observations be explained? The first reason is related to the specific properties of the operator; the dynamics of crossover are more difficult to analyze, as previous studies have shown [25]. Building a precise time series from a generated random walk is a hard task, when compared to mutation operators. The second reason is the dependence on parental distance. Larger parental distance usually induces larger crossover innovations [21]. Here, the crossover landscapes are generated from pairs of solutions where one of them is always created independently at random. This fact may ensure a larger parental distance, but at the same time, it might create a weak dependence on the distance between the parents, thus allowing a certain loss of homogeneity required to produce fitter offspring.

Representation must allow for crossover operators the capacity of efficient exchange of genetic material. For the MKP, we can observe that all studied representations clearly perform that task since they cover the feasible region of the search space. The exception is binary encoding. It can be concluded that the choice between these two crossover operators is not very influential on the representation used. These two operators, in conjunction with these representations, have the same behavior. There is no clear advantage of a single representation, which indicates that other factors must be taken into more consideration, e.g., population dynamics and selection.

D. Effect of Heuristics and Local Improvement

All decoder-based representations include some simple heuristic mechanisms that are essential for its interpretation. We will now complement our study by investigating a few other techniques that can be added to some of the encodings addressed within this research. Many additional techniques could be developed and used, but we restrain ourselves to the most common ones used for the MKP [21]. Some additional tests were made regarding the binary and the weight-coding representations. Our experimentation covers mutation and crossover-generated landscapes for these two representations. We only consider these two representations since these are the most common encodings with heuristics application found in the literature [21]. In our tests, we consider four improvements.

1) *Binary representation with repair (local method)* (BR with R): The standard encoding will be subject to a local method. This local method works by repairing unfeasible solutions by randomly removing items until all constraints are satisfied.
2) *Binary representation heuristic repair using profit/pseudoresource consumption ratios* (BR with HR): An unfeasible solution is repaired by removing items according to the order of the items given by the profit/pseudoresource consumption ratios.
3) *Binary representation with heuristic repair and local improvement* (BR with HR + LI): The union of the previous steps: a heuristic and a local method. In this case, the local method inserts items in a feasible solution, as long as it does not violate any constraint.
4) *Weight-coding representation with lognormal* (WC with lognormal): The heuristic is encoded in the mutation variation operator. The operator replaces a randomly selected position on the chromosome by sampling a random number that follows a lognormal distribution

$$w_j = (1+\gamma)^{N(0,1)}, \qquad j = 1, \ldots, n \tag{15}$$

where $\gamma > 0$ is a strategy parameter that controls the average intensity of biasing, and $N(0,1)$ denotes a normally distributed random number with a mean of 0 and a standard deviation of 1. We set $\gamma = 0.05$, as recommended by Raidl [20].

1) Mutation Landscapes: In Table III, we present, for all tested instances, the results of the applied measures for the four types of binary encodings and two types of weight-coding

TABLE III
SUMMARY RESULTS FOR THE EFFECT OF ADDING HEURISTICS AND LOCAL OPTIMIZATION FOR MUTATION LANDSCAPES

Representation	Instance			Measures			
	Name	n	m	$\overline{d_{opt}}$	ϱ	l	ξ
BR	P01	28	10	14.03 (0.36)	0.07	7.12	0.25
BR with R	P01	28	10	14.33 (0.38)	-0.39	9.93	0.35
BR with HR	P01	28	10	12.99 (0.43)	-0.58	12.46	0.44
BR with HR + LI	P01	28	10	1.78 (0.27)	-0.90	10.11	0.36
WC	P01	28	10	7.53 (0.34)	-0.58	14.51	0.52
WC with log-normal	P01	28	10	**0.19 (0.42)**	**-0.98**	**45.36**	**1.62**
BR	P02	50	5	25.01 (0.49)	0.09	16.18	0.32
BR with R	P02	50	5	26.13 (0.53)	-0.21	15.07	0.30
BR with HR	P02	50	5	23.93 (0.52)	-0.42	19.78	0.40
BR with HR + LI	P02	50	5	13.88 (0.17)	-0.84	18.28	0.37
WC	P02	50	5	17.48 (0.46)	-0.37	19.43	0.39
WC with log-normal	P02	50	5	**12.78 (0.31)**	**-0.86**	**75.17**	**1.50**
BR	CB01	100	5	49.95 (0.76)	-0.05	35.74	0.36
BR with R	CB01	100	5	50.37 (0.68)	-0.25	15.50	0.15
BR with HR	CB01	100	5	43.67 (0.87)	-0.43	21.83	0.22
BR with HR + LI	CB01	100	5	**6.58 (1.55)**	-0.96	23.79	0.24
WC	CB01	100	5	40.85 (0.69)	-0.62	10.85	0.11
WC with log-normal	CB01	100	5	9.62 (2.86)	**-0.99**	**138.31**	**1.38**
BR	CB02	100	5	50.00 (0.64)	-0.02	36.65	0.37
BR with R	CB02	100	5	49.94 (0.71)	-0.34	15.79	0.16
BR with HR	CB02	100	5	43.25 (0.83)	-0.47	19.73	0.20
BR with HR + LI	CB02	100	5	12.08 (1.11)	-0.91	12.77	0.13
WC	CB02	100	5	40.68 (0.74)	-0.59	10.78	0.11
WC with log-normal	CB02	100	5	**11.41 (2.60)**	**-0.98**	**133.35**	**1.33**
BR	GK01	100	15	49.95 (0.75)	0.06	27.32	0.27
BR with R	GK01	100	15	50.33 (0.70)	-0.06	15.64	0.16
BR with HR	GK01	100	15	49.80 (0.69)	-0.03	14.86	0.15
BR with HR + LI	GK01	100	15	**40.04 (0.44)**	**-0.98**	29.84	0.30
WC	GK01	100	15	49.41 (0.74)	-0.08	9.33	0.09
WC with log-normal	GK01	100	15	44.30 (0.80)	-0.61	**51.66**	**0.52**

(We include the results for simple BR and WC for comparison.) using mutation as the variation operator. Table IV contains the results for the crossover operators. A brief perusal of the results generated by mutation operators, as shown in Table III, reveals one important aspect: The weight-coding representation with the lognormal operator achieved the highest correlation value for all instances and the highest fitness distance correlation value for all instances, with the exception of instance GK01. At a closer examination, we can verify that the attained values are very good. For the fitness distance correlation, the encoding has values near -1.0 (instances P01, CB01, and CB02). Comparing these values to the average minimal distance from the best individual to the optimum, they are also the lowest. Only for instances P02 and Gk01 this effect is not so visible. Close to the weight-coding representation with lognormal operator is binary representation with consumption ratios, heuristic repair, and local improvement. This encoding is able to achieve very good values for the fitness distance correlation for all instances (In instance GK01, it has the best value.), as well as the average distance from the best individual to the optimum (instances CB01 and GK01). Regarding autocorrelation, this encoding does not attain the same degree of success as the weight-coding representation. (The values are similar to those attained by the simple direct encoding.)

From the examination of the fitness distance plots in Fig. 2, it is possible to distinguish two simple and clear patterns: 1) low correlated landscapes and 2) highly correlated landscapes. Binary representation with repair and binary representation with heuristic repair do not have a highly correlated

TABLE IV
SUMMARY RESULTS FOR THE EFFECT OF ADDING HEURISTICS AND LOCAL OPTIMIZATION FOR CROSSOVER LANDSCAPES

Representation	Instance			One point Crossover			Uniform Crossover		
	Name	n	m	$\overline{d_{opt}}$	ϱ_{cx}	ρ_{cx}	$\overline{d_{opt}}$	ϱ_{cx}	ρ_{cx}
BR	P01	28	10	13.23 (0.36)	-0.04	0.72	13.46 (0.37)	-0.03	0.54
BR with R	P01	28	10	13.08 (0.36)	-0.40	0.92	13.19 (0.36)	-0.40	0.85
BR with HR	P01	28	10	10.04 (0.39)	-0.64	**0.96**	11.04 (0.40)	-0.60	**0.90**
BR with HR + LI	P01	28	10	**3.63** (0.33)	**-0.68**	0.54	**7.00** (0.38)	**-0.67**	0.72
BR	P02	50	5	24.53 (0.51)	0.01	0.78	24.65 (0.53)	0.01	0.65
BR with R	P02	50	5	25.14 (0.52)	-0.29	0.82	25.25 (0.49)	-0.28	0.81
BR with HR	P02	50	5	22.02 (0.50)	-0.35	**0.90**	22.28 (0.45)	-0.35	**0.88**
BR with HR + LI	P02	50	5	**15.04** (0.40)	**-0.56**	0.65	**16.08** (0.47)	**-0.51**	0.65
BR	CB01	100	5	49.64 (0.72)	-0.02	0.80	49.74 (0.71)	-0.01	0.75
BR with R	CB01	100	5	48.19 (0.69)	-0.41	0.75	48.36 (0.70)	-0.42	0.71
BR with HR	CB01	100	5	39.90 (0.83)	-0.53	**0.84**	40.04 (0.81)	-0.52	**0.81**
BR with HR + LI	CB01	100	5	**33.81** (0.84)	**-0.62**	0.77	**36.34** (0.78)	**-0.58**	0.72
BR	CB02	100	5	49.62 (0.71)	-0.03	0.82	49.72 (0.71)	-0.03	0.75
BR with R	CB02	100	5	47.85 (0.68)	-0.41	0.76	48.01 (0.69)	-0.42	0.73
BR with HR	CB02	100	5	39.26 (0.84)	-0.54	**0.86**	39.83 (0.83)	-0.53	**0.82**
BR with HR + LI	CB02	100	5	**34.26** (0.86)	**-0.64**	0.78	**36.30** (0.82)	**-0.61**	0.72
BR	GK01	100	15	49.89 (0.71)	-0.02	0.78	49.91 (0.71)	0.01	0.72
BR with R	GK01	100	15	50.38 (0.71)	-0.05	0.77	50.35 (0.72)	-0.03	0.72
BR with HR	GK01	100	15	49.66 (0.67)	-0.06	**0.81**	49.50 (0.66)	-0.06	**0.75**
BR with HR + LI	GK01	100	15	**48.40** (0.71)	**-0.10**	0.72	**48.73** (0.69)	**-0.12**	0.66

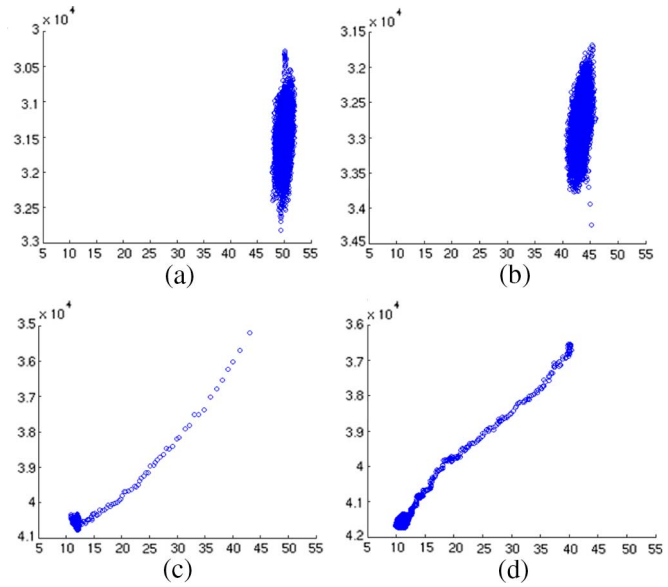


Fig. 2. Fitness distance correlation plots for (a) binary representation with repair, (b) binary representation with heuristic repair, (c) binary representation with heuristic repair and local improvement, and (d) weight coding with lognormal operator on the CB02 instance. The graphs plot the *distance to the optimum* versus the *fitness value*.

landscape, whereas binary representation with heuristic repair and local improvement and weight-coding representation with the lognormal operator have highly correlated landscapes. These representations have near-ideal distributions since the solutions very close to the optimum can be reached by *jumping* from one solution to another by successively reducing the *jump* distance. For these two encodings, the addition of heuristics and/or local improvement was influential in reshaping the distribution shapes (for binary representation in one configuration) in comparison to the distributions of the representations without heuristics and/or local improvement. Instead of ellipsoidal shapes, the distributions are now closer to linear lines in the direction of the optimum, with a strong concentration on that point. Fitness and distance are clearly correlated.

This analysis reveals that adding heuristics and/or local improvement methods to a representation is crucial in obtaining better results when solving the MKP. Examining the weight-coding representation, using a mutation variation operator that is more sensitive to the problem domain helps in improving performance. Since the float values represent surrogate multipliers, it is clear that the mutation provided by a uniform distribution will cause strong phenotypic changes. On the other hand, a lognormal distribution will handle the genotype in a more subtle way since it gives the advantage of making small changes in weight with higher probabilities and allowing large changes but with less probability. The lognormal operator ensures a higher locality for the weight-coding representation. In this particular case, the addition of a heuristic—by means of a more controlled probability distribution—helped the encoding to improve its fitness distance correlation and autocorrelation values for all tested problem instances.

The binary representation needs a more careful investigation. The introduction of a local search method was sufficient in improving the measured values when compared to the simple direct encoding (as shown in [4] and [20]) but was unable to achieve the values of the ordinal, permutation, and random-key representations. These three encodings still perform better than the direct encoding even when we compare them to the binary representation with the heuristic ratios (BR with HR). The exception is instance P01, although the differences are small. It seems that these two improvements, when examined separately from each other, can only introduce a slightly weaker heuristic bias than the first-fit heuristic used for ordinal, permutation, and random-key encodings. When heuristic repair and local improvement are combined, the result is very different, and the behavior is comparable to the weight-coding representation with the lognormal mutation operator. In terms of autocorrelation, the effects are different. From Table III, we can see that the simple direct encoding is poor due to the unfeasibility of most

generated solutions, whereas this changes for the other remaining variants, because they ensure feasibility. In addition, the difference between the three types of additional mechanisms added to the encodings is evident. The combination of a local method and a heuristic is better than their single use. For all variants of the direct encoding, the autocorrelation values are similar, with only marginal differences.

2) Crossover Landscapes: We proceed now to the crossover-generated landscape analysis. In this case, only the modifications made to the simple binary encoding are eligible. For weight-coding representation, the heuristic was introduced on the mutation operator, and the same heuristic is not directly applicable to the crossover operator. Focusing our attention on the results from Table IV, we can see that, for crossover-generated landscapes, the fitness distance correlation and the minimum average distance to the optimum solution, the direct encoding with the heuristic repair, and the local improvement method attain the best results. These results confirm the previous observations. Usually, crossover has a role of exploration. The addition of mechanisms that enable some exploitation when evaluating the individuals allows finding better solutions. In spite of this, the degree of improvement obtained by introducing the combined use of these two extra mechanisms (heuristics and local improvement) is not so large in comparison with mutation. The interesting fact is that the sole use of these mechanisms might be more sensitive to the instance's structure. For the instances of the first benchmark, heuristic repair performs better than simple repair, whereas, for the second one, simple repair gives better results. The type of crossover does not affect the representation's performance.

In terms of parent–offspring correlation, results are more consistent when compared with mutation. Tested encodings reveal higher values of correlation with some minor exceptions: binary representation with uniform crossover for the first instance P01 (0.54) and binary representation with heuristic repair and local improvement on both operators (0.65). For all other experiments, correlation values are high within in the range of [0.7, 0.9]. This can be explained by the strong heuristic bias that is introduced by binary encodings. As already seen with mutation, the binary encoding permits good correlation values since alterations to a candidate solution generate neighbor solutions, which are not only close but, with the addition of these mechanisms, whose variations on the fitness values are also small. As mentioned before, in terms of correlation, the type of crossover has no influence on the encoding behavior.

As we can see from these results, a pattern arises in terms of the representation performances according to their heuristic bias. This is essentially true when looking at the fitness distance correlation coefficient. The representations with stronger heuristic bias (weight-coding with lognormal operator and binary representation with heuristic repair and local improvement) achieve the best results, i.e., very close to -1.0. The encodings with a weak heuristic bias do perform well but clearly worse than the two previous encodings, whereas the simple binary representation, as expected, is the one with the poorest behavior. Our fitness landscape analysis is in accordance with previous studies on representations for the MKP, such as [21].

TABLE V
SUMMARY OF OPTIMIZATION RESULTS FROM [21]

Instance			Representations				
Name	n	m	OR	PR	RK	BR+HR+LI	WC+log
CB2	250	5	1.321	0.120	0.115	0.106	0.106
			(0.346)	(0.012)	(0.009)	(0.006)	(0.007)
CB5	250	10	1.498	0.295	0.277	0.249	0.261
			(0.225)	(0.033)	(0.021)	(0.017)	(0.008)
CB8	250	30	2.076	0.608	0.611	0.535	0.519
			(0.346)	(0.048)	(0.072)	(0.031)	(0.013)
CB3	500	5	2.382	0.081	0.065	0.038	0.042
			(0.657)	(0.016)	(0.010)	(0.008)	(0.003)
CB6	500	10	2.815	0.225	0.200	0.112	0.131
			(0.462)	(0.040)	(0.029)	(0.014)	(0.007)
CB9	500	30	3.267	0.377	0.376	0.288	0.306
			(0.442)	(0.058)	(0.037)	(0.024)	(0.012)
GK05	200	25	1.124	0.462	0.552	0.294	0.397
			(0.153)	(0.072)	(0.118)	(0.046)	(0.004)
GK06	200	50	1.236	0.703	0.751	0.611	0.429
			(0.141)	(0.070)	(0.108)	(0.060)	(0.018)
GK07	500	25	1.468	0.523	0.651	0.382	0.093
			(0.092)	(0.088)	(0.087)	(0.082)	(0.004)
GK08	500	50	1.517	0.749	0.835	0.534	0.166
			(0.109)	(0.086)	(0.125)	(0.066)	(0.006)
GK09	1500	25	2.312	0.890	1.064	0.029	0.558
			(0.113)	(0.075)	(0.133)	(0.042)	(0.001)
GK10	1500	50	1.883	1.101	1.177	0.052	0.727
			(0.076)	(0.065)	(0.082)	(0.070)	(0.003)
GK11	2500	100	1.677	1.237	1.246	0.052	0.867
			(0.056)	(0.060)	(0.067)	(0.061)	(0.002)

E. Optimization Results and Landscape Analysis

The landscape analysis provides some insights about the behavior of these different genetic representations during an evolutionary process. However, it is important to relate this with results obtained by optimization runs. The MKP has been studied in the context of optimization. In [21], a study is provided with results from the optimization on the MKP using current state-of-the-art evolutionary algorithms for this problem and the genetic representations described in this paper. We will use these optimization results to find some connections between landscape analysis and optimization for the MKP. Table V contains the largest instances from the two benchmarks used in our study: 1) the *OR-Library* (250 and 500 items) and 2) the suite provided by Glover and Kochenberg (200, 500, 1500, and 2500 items). We use the instance names provided by Raidl and Gottlieb [21]. Notice that Table V only contains results for the ordinal, permutation, and random-key encodings; weight-coding with lognormal mutation; and binary encoding with heuristic and local improvement. The values for each encoding represent the mean best fitness found in the runs in terms of *gap*, i.e., the distance between the best value found and the optimum obtained through an LP-relaxed problem (the brackets include the standard deviation). For the complete results on all tested instances, we refer the reader to [21].

In Table V, it can be seen that binary representation with local improvement and heuristic is the best approach, followed by weight-coding with lognormal mutation. For the first benchmark, these two representations attain a similar performance, which is not seen in the second benchmark. The remaining encodings perform more or less on the same level, with the exception of ordinal representation. When looking at the results, for the larger and more difficult instances of both benchmarks,

the main conclusions obtained with the fitness landscape analysis are confirmed: Binary representation with heuristics and local improvement, and weight-coding with the lognormal mutation operator are the best representations for this problem. Nevertheless, optimization results suggest that direct encoding with auxiliary mechanisms is the best representation. It should be noticed that, for most of the instances, the performance is similar and that other factors also play a role in these results, such as the dynamics of the algorithm. From the results attained by Raidl and Gottlieb [21], direct encoding seems more suitable for the second benchmark suite, whereas, for the first one, both encodings are appropriate. This can also be seen in the results of Table III, where weight-coding and binary encoding have the same kind of behavior on the first suite, and on the second, direct encoding clearly shows better values for fitness distance correlation.

Our analysis focused on smaller and easier instances of the MKP (which are solved by exact methods) since this allows us to circumvent one of the problems of fitness landscape analysis—how to estimate the distance from the closest global optimum, if the global optima are unknown. In spite of having some interest in running this same analysis on larger instances, optimization results show that our findings can be related to the performance of the encodings.

VI. Conclusion

In this paper, we presented a fitness landscape analysis for the MKP. The goal of this paper was to study how the interplay between representations, heuristics, and genetic operators affects the search performance of evolutionary algorithms for the MKP. Although it is a relevant topic, only a few studies that analyze this issue exist.

Standard tools of fitness landscape analysis, such as fitness distance correlation and autocorrelation, help in explaining the differences in performance achieved by different representations. Within a mutation-based evolutionary algorithm, weight-coding with a lognormal mutation operator and binary representation with a heuristic repair and local improvement appear to be the most suitable combinations. Considering crossover-based algorithms, the binary representation with the heuristic repair and local improvement is the suitable choice. In addition, the study described in this paper presents a contribution to analyze how heuristics and local search techniques can improve the performance of evolutionary algorithms. In general, heuristics with a strong bias help in achieving better results. Representation plays an important role when solving the MKP since choosing an encoding without a strong heuristic bias can create some difficulties for the evolutionary algorithm. As such, the use of heuristics on the decoding process or on a variation operator, as well as the use of local improvement methods, can significantly alter the performance for a given representation.

Even though our analysis focused on the MKP, the aim of our research is to investigate the influence of genetic representations and heuristics when solving combinatorial optimization problems. This research will now be extended to different combinatorial optimization problems to see if the findings encountered here can be generalized. The outcomes of the global analysis may be important for future applications of evolutionary algorithms to problems with similar properties.

References

[1] K. Deb, "Introduction to representations," in *Evolutionary Computation 1: Basic Algorithms and Operators*, T. Bäck, D. B. Fogel, and Z. Michalewicz, Eds. Bristol, U.K.: Inst. Phys. Publishing, 2000, ch. 14, pp. 127–131.

[2] F. Rothlauf, *Representations for Genetic and Evolutionary Algorithms*. New York: Springer-Verlag, 2002.

[3] S. Wright, "The roles of mutation, inbreeding, crossbreeding and selection in evolution," in *Proc. VI Int. Conf. Genetics*, 1932, vol. 1, pp. 356–366.

[4] P. Chu and J. Beasley, "A genetic algorithm for the multidimensional knapsack problem," *J. Heuristics*, vol. 4, no. 1, pp. 63–86, 1998.

[5] J. Tavares, F. B. Pereira, and E. Costa, "The role of representation on the multidimensional knapsack problem by means of fitness landscape analysis," in *Proc. Congr. Evol. Comput.*, Jul. 16–21, 2006, pp. 2307–2314.

[6] S. Martello and P. Toth, *Knapsack Problems: Algorithms and Computer Implementations*. New York: Wiley, 1990.

[7] H. Kellerer, U. Pferschy, and D. Pisinger, *Knapsack Problems*. New York: Springer-Verlag, 2004.

[8] J. Gottlieb, "Evolutionary algorithms for constrained optimization problems," Ph.D. dissertation, Tech. Univ. Clausthal, Clausthal-Zellerfeld, Germany, 1999.

[9] G. R. Raidl, "An improved genetic algorithm for the multiconstrained 0–1 knapsack problem," in *Proc. 5th IEEE Conf. Evol. Comput., IEEE World Congr. Comput. Intell.*, Anchorage, AK, May 1998, pp. 207–211.

[10] Z. Michalewicz and J. Arabas, "Genetic algorithms for the 0/1 knapsack problem," in *Proc. 8th ISMIS*, 1994, pp. 134–143.

[11] R. Hinterding, "Mapping order-independent genes and the knapsack problem," in *Proc. 1st IEEE Int. Conf. Evol. Comput.*, 1994, pp. 13–17.

[12] J. Thiel and S. Voss, "Some experiences on solving multiconstraint zero–one knapsack problems with genetic algorithms," *INFOR*, vol. 32, no. 4, pp. 226–242, 1994.

[13] J. Gottlieb, "Permutation-based evolutionary algorithms for multidimensional knapsack problems," in *Proc. ACM SAC*, 2000, pp. 408–414.

[14] J. Gottlieb, "On the feasibility problem of penalty-based evolutionary algorithms for knapsack problems," in *Proc. EvoWorkshops Appl. Evol. Comput.*, 2001, pp. 50–59.

[15] J. Gottlieb and G. R. Raidl, "Characterizing locality in decoder-based EAs for the multidimensional knapsack problem," in *Proc. Sel. Papers 4th Eur. Conf. AE*, 2000, pp. 38–52.

[16] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, 3rd ed. Berlin, Germany: Springer-Verlag, 1992.

[17] J. C. Bean, "Genetic algorithms and random keys for sequencing and optimization," *ORSA J. Comput.*, vol. 6, no. 2, pp. 154–160, 1994.

[18] R. Hinterding, "Representation, constraint satisfaction and the knapsack problem," in *Proc. Congr. Evol. Comput.*, P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, and A. Zalzala, Eds., Jul. 6–9, 1999, vol. 2, pp. 1286–1292.

[19] C. Cotta and J. Troya, "A hybrid genetic algorithm for the 0–1 multiple knapsack problem," in *Artificial Neural Nets and Genetic Algorithms 3*, G. Smith, N. Steele, and R. Albrecht, Eds. New York: Springer-Verlag, 1998, pp. 251–255.

[20] G. Raidl, "Weight-codings in a genetic algorithm for the multiconstraint knapsack problem," in *Proc. Congr. Evol. Comput.*, P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, and A. Zalzala, Eds., Jul. 6–9, 1999, vol. 1, pp. 596–603.

[21] G. R. Raidl and J. Gottlieb, "Empirical analysis of locality heriability and heuristic bias in evolutionary algorithms: A case study for the multidimensional knapsack problem," *Evol. Comput. J.*, vol. 13, no. 4, pp. 441–475, Dec. 2005.

[22] B. Manderick, M. de Weger, and P. Spiessens, "The genetic algorithm and the structure of the fitness landscape," in *Proc. 4th Int. Conf. Genetic Algorithms*, 1991, pp. 143–150.

[23] T. Jones, "Evolutionary Algorithms, Fitness landscapes and search," Ph.D. dissertation, Univ. New Mexico, Albuquerque, NM, May 1995.

[24] C. R. Reeves, "Fitness landscapes and evolutionary algorithms," in *Proc. Sel. Papers 4th Eur. Conf. AE*, 1999, pp. 3–20.

[25] P. Merz, "Memetic algorithms for combinatorial optimization problems: Fitness landscapes and effective search strategies," Ph.D. dissertation, Univ. Siegen, Siegen, Germany, 2000.

[26] W. Hordijk and B. Manderick, "The usefulness of recombination," in *Proc. 3rd Eur. Conf. Artif. Life*, 1995, pp. 908–919.

[27] L. Altenberg, "Fitness distance correlation analysis: An instructive counterexample," in *Proc. 7th ICGA*, 1997, pp. 57–64.
[28] B. Sendhoff, M. Kreutz, and W. V. Seelen, "A condition for the genotype-phenotype mapping: Casualty," in *Proc. 7th Int. Conf. Genetic Algorithms*, 1997, pp. 73–80.

Jorge Tavares (S'04–M'08) received the B.Sc., M.S., and Ph.D. degrees in informatics engineering from the University of Coimbra, Coimbra, Portugal, in 2002, 2003, and 2007, respectively.

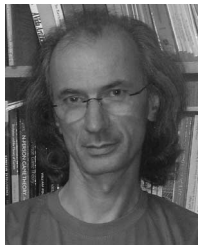
From 2002 to 2007, he was with the Center for Informatics and Systems of the University of Coimbra, Department of Informatics Engineering, University of Coimbra, where he was a Researcher and a Member of the Evolutionary and Complex Systems Group. He is currently with the French National Institute for Research in Computer Science and Control (INRIA), Research Center Lille-Nord Europe, Lille, France, as a Postdoctoral Researcher and a member of the DOLPHIN Team. He has regularly published in the proceedings of international conferences and workshops related to evolutionary computation. His research interests are mainly optimization by means of biological inspired techniques, design and analysis of representations for evolutionary algorithms, and self-organized evolutionary systems.

Dr. Tavares is a member of the Association for Computing Machinery. He has served as a member of program committees and reviewer for several international conferences.

Francisco B. Pereira (M'05) received the B.Sc. degree from the University of Coimbra (UC), Coimbra, Portugal, in 1994, the M.S. degree from the Universidade Nova de Lisboa, Lisboa, Portugal, in 1995, and the Ph.D. degree from UC, in 2002, all in computer science.

He is currently an Assistant Professor with the Department of Computer Science, Instituto Superior de Engenharia de Coimbra (ISEC), Polytechnic Institute of Coimbra, Coimbra. He is also a Researcher with the Center for Informatics and Systems of University of Coimbra, Department of Informatics Engineering, University of Coimbra. He has published more than 30 peer-reviewed articles. His research interests are the application of evolutionary computation techniques to optimization problems.

Prof. Pereira has, in the past few years, organized some international workshops that are related to evolutionary computation and has been a regular member of the program committee of several international conferences. He was the recipient of two Best Paper Awards.



Ernesto Costa received the B.Sc. degree in electronics engineering from the University of Coimbra (UC), Coimbra, Portugal, in 1976, a 3rd Cycle Thesis in Computing Science from the University Pierre et Marie Curie, Paris, France, in 1981, and the Ph.D. degree in electronic engineering, in the area of computing science, from UC, in 1985.

He was a Cofounder of the Center for Informatics and Systems of the University of Coimbra (CISUC) and its President between May 1998 and June 2000. He was also the Founder of the Artificial Intelligence Group, which he led until 2003, when he founded the new Evolutionary and Complex Systems Group. Since 2005, he has been a member of the Scientific Advisory Board of SolveIT Software. He is currently a Full Professor and the Head of the Scientific Board of the Department of Informatics Engineering, UC. He has published more than 100 papers in books, journals, and conference proceedings. His research interests are evolutionary computation and applications of artificial intelligence to the real world.

Prof. Costa has participated in several projects and organized several international scientific events. He was the recipient of three international prizes.