The Role of Representation on the Multidimensional Knapsack Problem by means of Fitness Landscape Analysis

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

***Abstract*— Five encodings for the Multidimensional Knapsack Problem are investigated, using fitness landscape analysis techniques, in order to better understand the influence of genetic representations when solving a combinatorial optimization problem. Fitness distance correlation and autocorrelation measures are employed to analyze the encodings. The effect of heuristics, as well as repair and local optimization is also examined. The investigation helps to understand how the adopted representations influence the search performance of an evolutionary algorithm.**

I. Introduction

Evolutionary Algorithms (EA) have been successfully applied to combinatorial optimization problems over the past few years. Nevertheless, most of the research deals with applying an EA to a new problem or applying a new EA to an existing one. In most of these studies, the focus is on showing *how* the EA was able to solve the problem (or find better solutions) and not in *why* it was able to solve it. Since fitness landscape analysis techniques were introduced (consult [1] for a detailed work on fitness landscape analysis), they have become a valuable tool to analyze and investigate why a given EA works. They have been applied to many different problems providing understanding on the *whys* of evolutionary computation approaches.

Different applications of EAs [2] exist for the Multidimensional Knapsack Problem (MKP). The choice of a suitable representation plays a crucial role on the design of an EA. For the MKP, several representations exist and the aim of this paper is to investigate different encodings, as well as the effect of adding heuristics and local search. To the best of our knowledge, this is the first study regarding fitness landscapes analysis on the MKP.

The paper is structured as follows: a formal definition of the MKP is presented in section II; section III presents an overview of related work; in section IV we describe the fitness landscape analysis techniques; experimental results and discussion are reported in section V; finally, in section VI we present some conclusions.

II. The Multidimensional Knapsack Problem

The MKP is a NP-Hard combinatorial optimization problem, with a wide range of applications such as cargo loading, cutting stock and budget management [3]. This problem is also known in the literature as the *M-Dimensional Knapsack Problem*, the *Multiconstraint Knapsack Problem*, the *Multi-Knapsack Problem* and the *Multiple Knapsack Problem*. In some literature, authors also include in their name the term *zero-one*. Using alternative names for the same problem is potentially confusing. Since the majority of the authors appear to have used the *Multidimensional Knapsack Problem* designation [2], we adopt the same name.

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, it is stated as:

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

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

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

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

The decision variable is the binary vector $x = (x_1, \ldots, x_n)$. Each item j is mapped to a bit. When $x_k = 1$, the corresponding item is considered part of the solution. The special case of $m = 1$ is generally known as the *Knapsack Problem*, and is a solvable in pseudo-polynomial time (it is only weakly NP-Hard).

The MKP has been widely studied for 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 [3] or [4].

III. Evolutionary Approaches

Evolutionary approaches have shown to be effective for searching and finding good quality solutions for the MKP. The most successful EAs are based on repairing and local optimization [2],[5]. The majority of the proposed approaches ([2], [6], [7], [8] and [9]) can be divided in two categories according to the adopted representation and the associated variation operators. Algorithms may use a direct representation, encoded as a binary string, where each bit is mapped to an item. A bit set to 1 indicates that the corresponding item is packed into the knapsack. Moreover, EAs may adopt indirect encodings, such as permutations or weight-codings. In this case, the algorithm needs a decoder that translates the chromosome into the actual solution. The choice of a

Jorge Tavares and Ernesto Costa are with the Centre for Informatics and Systems of the University of Coimbra, Pólo II - Pinhal de Marrocos, 3030 Coimbra, Portugal (emails: {jast, ernesto}@dei.uc.pt).

Francisco B. Pereira is also with the Polytechnic Institute of Coimbra, Quinta da Nora, 3030 Coimbra, Portugal (email: xico@dei.uc.pt).

representation is a critical issue in the design of an EA for the MKP. Although a binary string is the most natural encoding, it generates infeasible solutions making the task for an EA harder. Instead, indirect encodings focus the search of the EA on the boundary of the feasible region [10], suggesting that they might be the most appropriate representations. Anyway, this is not a straightforward decision, as confirmed by several studies [11].

Next, we briefly describe the most common representations for the MKP and a simple description of some significant works regarding EAs and MKP.

A. Genetic Representations

1) Binary Representation: With this encoding a solution is represented by a characteristic bit vector. There are two options to deal with infeasible solutions: penalty-based fitness functions [12], or repair operators. Chu and Beasley [2] proposed a repair mechanism, that iteratively removes items until all constraints are satisfied. In addition, each solution is improved by local optimization. Both methods are guided by a heuristic. This approach was later improved by Raidl [5], by using the heuristic associated with the weight-coding representation, described later.

2) Ordinal Representation: 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; next, vector v is traversed in order and each v_k specifies a position in L; the referenced element is removed from L and is inserted in permutation π. To decode the permutation into a feasible MKP solution, a *first-fit* heuristic is applied. It builds a feasible solution by traversing all variables in the order given by the permutation π. An item π_j is inserted into the solution if it does not violates any constraint. This representation has the advantage to enable the use of standard genetic operators. EAs using this representation can be found in [6] and [13]. These studies report poor results in comparison with other representations.

3) Permutation Representation: Permutations have been widely applied to this problem [5], [7], [8], [9]. The representation consists of a permutation of all items, $\pi = (\pi_1, \ldots, \pi_n)$. 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 and swap mutation. Permutation-based EAs have achieved good results when applied to the MKP [9].

4) Random-Key Representation: The random-key representation is an alternative approach to encode permutations without the need for specific operators [14]. A random-key is a vector of real values $w = (w_1, \ldots, w_n)$ where each position represents an item j, 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 weights indexes. Once again, the attained permutation is decoded by means of a *first fit* heuristic. Evolutionary approaches based on random-keys for MKP can be found in [7] and [15]. The reported results show that EAs based on this representation can achieve good results.

5) Weight-Coding Representation: The weight-coding representation is the most successful decoder-based technique [16]. In this case, a chromosome consists of a real-valued vector $w = (w_1, \ldots, w_n)$ where each item j is associated with a weight $w_j \in [0, 1]$. The decoding process consists of transforming the original problem P into a modified problem $P^{'}$, by multiplying the original items profits with the associated weight. 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. The heuristics used 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 constraints into a single one:

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

where a_i 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., the 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/pseudo-resource consumption ratios*, u_j, are calculated as:

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

According to the u_j values, we sort the items in a decreasing order, adding them to the solution one at a time if none of the constraints are violated. The surrogate multipliers are only calculated once for the original problem at the begining. This is done to ensure low computation costs and as a result of this step, the decoding process begins by determining the u_j values. For details of this method, please refer to [16]. This approach has attained very good results when applied to the MKP.

B. Analysis of Evolutionary Algorithms for the MKP

Although many different EAs have been applied to the MKP, the number of studies that try to perform a comprehensive analysis of the behavior of such algorithms is limited. The most significant work on this subject was done by Raidl and Gottlieb [11]. In this study, five representations are examined according to important aspects in a EA, such as initialization and variation operators. Three properties, locality, heritability and heuristic bias, are considered. A set of static measures is used to gain insights about the EAs behavior. This investigation helped to confirm and extend previous studies about the MKP, such as [9], [13], [17] and [18]. Another important study was performed by Branke et al. [19]. In this work, a dynamic version of the MKP is proposed and analyzed according to the changing fitness landscape.

This work is a contribution to a better comprehension of EAs in dynamic environments.

In this paper, different representations for the MKP are studied using fitness landscape analysis. This investigation aims to complement previous studies and contribute to a clear understanding of how a representation can influence the performance of an EA when searching for good solutions for the MKP.

IV. Fitness Landscapes

The concept of *fitness landscape*, introduced by Wright [20], has been useful for the analysis and understanding of the behavior of EAs. Furthermore, the study of fitness landscapes can be of value in designing an EA, since it can help in predicting its performance. Fitness landscapes can be defined as follows: a set of points (solutions) X, a fitness function f which assigns a numeric value to each solution, and a distance operator d defining the neighborhood within set X. Formally, a fitness landscape is a tuple $\mathcal{L} = (X, f, d)$ with a fitness function defined as $f : X \rightarrow \Re$. Thus, the fitness landscape can be interpreted as a graph $G_{\mathcal{L}} = (X, E)$ with edge set $E = \{(x, y) \in X \times X \mid d(x, y) = d_{min}\}$, with d_{min} being the minimum distance between two points in the search space. The diameter of a landscape $diamG_{\mathcal{L}}$, is the maximum distance in the search space. As an example, for binary representations with a flip operator, the solution set is $X = \{0, 1\}^n$, with $G_{\mathcal{L}}$ a hypercube of dimension n and the Hamming distance as the distance operator. This yields $d_{min} = 1$ and $diamG_{\mathcal{L}} = n$. For a comprehensive study of fitness landscapes for EAs and combinatorial optimization problems, please refer to the works of Jones [1] and Merz [21].

A. Landscapes Analysis Techniques

The structure of fitness landscapes influences the ability of an EA to perform an efficient search. 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 and the landscape ruggedness. These characteristics are well known and have been studied in the evolutionary computation community (see, e.g. [21] for an overview).

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

$$\varrho(f, d) \approx \frac{1}{\sigma(f)\sigma(d)m} \sum_{i=1}^{m} (f_i - \overline{f})(d_i - \overline{d}) \qquad (7)$$

with a given set of points x_i of size m (the random walk length) and $f_i = f(x_i)$ the fitness value and $d_i = d_{opt}(x_i)$ the minimum distance to a global optimum solution. The $\overline{f}$ and $\sigma(f)$ are the mean and standard deviation. 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. A value of $\varrho = 1.0$ indicates the opposite.

2) The 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 an EA 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 being harder to search in it. To measure the ruggedness of a fitness landscape we calculate the *autocorrelation function* $\rho(d)$, which measures the correlation of all pairs of points in the search space with distance d. It can be estimated by computing a large sample of solutions:

$$\rho(d) = \frac{1}{\sigma^2(f)|X^2(d)|} \sum_{(x,y)\in X^2(d)} (f(x) - \overline{f})(f(y) - \overline{f}) \quad (8)$$

where $X^2(d)$ is the set of all pairs of points in the search space with distance d and $|X^2|$ the number of pairs in the set. Another possibility to estimate $\rho(d)$ is to perform a random walk. In this case, the random walk of a time series $\{f(x_t)\}$ defines the correlation of two points s steps away in a m length random walk:

$$\rho_{rw}(s) \approx \frac{1}{\sigma_f^2(m-s)} \sum_{t=1}^{m-s} (f(x_t) - \overline{f})(f(x_{t+s}) - \overline{f}) \quad (9)$$

If the landscape is *statistically isotropic*, the time series forms a stationary random process, therefore a single random walk is sufficient to obtain ρ_{rw} [21].

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

$$l = -\frac{1}{ln(|\rho(1)|)} \qquad (10)$$

for $\rho(1) \neq 0$. It is useful to normalize it with the diameter of the landscape:

$$\xi = \frac{l}{diamG_{\mathcal{L}}} \qquad (11)$$

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.

V. Analysis of Landscapes

In this section, we analyze the influence of representations when solving the MKP. This analysis is performed by conducting an investigation on the fitness landscapes defined by different representations. We consider five representations: binary, ordinal, permutation, random-key and weight-coding. Each one of them has associated a standard

mutation operator. These are: bit-flip mutation, integer flip mutation, swap and uniform flip mutation. For the weight-coding representation, we will first use the uniform flip operator and later we will switch to a different one. This is done to examine the influence of the operator heuristic bias. All representations cover the feasible region of the search space with the exception of the direct encoding. As such, we use a penalty function:

$$penalty(x) = \frac{p_{max} + 1}{r_{min}} \times max\{CV(x,i) \mid i \in I\} \quad (12)$$

Where p_{max} is the maximum profit p_i and r_{min} is the minimum resource consumption $r_{i,j}$, with $I = \{1, \ldots, m\}$ and $J = \{1, \ldots, n\}$. CV is calculated by:

$$CV(x,i) = max(0, \sum_{j \in J} r_{ij} x_j - c_i) \quad (13)$$

This penalty function is recommended since it can better guide the search to the feasible region [12].

In this study we consider the binary representation in different forms. First, the simple binary encoding with penalty-based fitness function, then we will add the *profit/pseudo-resource consumption ratios* to obtain a better solution, and finally, we will add repair and local optimization. This is done to better examine the effect of heuristics and local search on the encoding. We begin our analysis on standard and simple encodings that do not use heuristics particularly biased towards fitter phenotypes [11].

A. Experimental Setup

For our analysis, we selected some 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]. From the first suite, we used the instances containing 28 and 50 items, with 10 and 5 constraints respectively (designated P01 and P02). From the second dataset, we selected instances with 100 items and 5 constraints and 25% of tightness (designated CB01 and CB02). From the last benchmark suite, we chose the instance with 100 items and 15 constraints (named GK01). A fitness distance analysis requires that the global optima are known. For these instances we solved them to optimality by running a MIP solver from the GNU Linear Programming Kit [22]. For each one of the random walks, we performed 50 runs with 100000 steps. We performed more tests with different problem instances, all showing the same trends presented in this work.

B. Fitness Distance Correlation

In table I we present, for all five MKP instances, the results of the applied measures. The first column indicates the representation used (BR - binary representation, OR - ordinal representation, PR - permutation representation, RK - random-key representation, WC - weight-coding representation), the column with n indicates the number of items and m the number of resources. The average of the minimum distance between the optimum solution and the best found individual in the random walk can be observed on the next column ($\overline{d_{opt}}$) with the standard deviation in brackets. The last column presents values for the fitness distance correlation (ϱ).

TABLE I
SUMMARY RESULTS FOR THE FITNESS DISTANCE CORRELATION ON ALL THE MKP INSTANCES FOR THE FIVE REPRESENTATIONS.

Representation	Instances	n	m	$\overline{d_{opt}}$	ϱ
BR	P01	28	10	14.03 (0.36)	0.07
OR	P01	28	10	12.02 (0.35)	-0.51
PR	P01	28	10	12.01 (0.35)	-0.51
RK	P01	28	10	12.03 (0.36)	-0.53
WC	P01	28	10	**7.53 (0.34)**	**-0.58**
BR	P02	50	5	25.01 (0.49)	0.09
OR	P02	50	5	20.67 (0.57)	**-0.45**
PR	P02	50	5	20.64 (0.53)	-0.41
RK	P02	50	5	20.72 (0.54)	-0.28
WC	P02	50	5	**17.48 (0.46)**	-0.37
BR	CB01	100	5	49.95 (0.76)	-0.05
OR	CB01	100	5	49.70 (0.71)	-0.57
PR	CB01	100	5	49.64 (0.71)	-0.53
RK	CB01	100	5	49.63 (0.72)	-0.56
WC	CB01	100	5	**40.85 (0.69)**	**-0.62**
BR	CB02	100	5	50.00 (0.64)	-0.02
OR	CB02	100	5	49.78 (0.71)	**-0.60**
PR	CB02	100	5	49.81 (0.70)	-0.54
RK	CB02	100	5	49.64 (0.67)	-0.53
WC	CB02	100	5	**40.68 (0.74)**	-0.59
BR	GK01	100	15	49.95 (0.75)	0.06
OR	GK01	100	15	49.99 (0.70)	-0.15
PR	GK01	100	15	49.99 (0.77)	-0.12
RK	GK01	100	15	49.91 (0.77)	**-0.19**
WC	GK01	100	15	**49.41 (0.74)**	-0.08

The data present in table I shows that the average distances between the best found individual and the optimal solution, for the first two instances (P01 and P02), are similar for the ordinal, permutation and random-key encodings. Binary representation has the worst performance while weight-coding has the best results. In terms of percentage, the binary encoding is always around a value of 50% from the optimum, whilst the ordinal, permutation and random-key representations are around 42%. To complete, weight-coding is on the interval of 26% to 35%. This pattern is not found on the larger instances. For all the instances with $n = 100$, the average distance to the optimum solution is very similar for all tested representations, around the value of 50%. The exception is weight-coding which performs slightly better on instances CB01 and CB02 but on the same level as the other

[1]http://mscmga.ms.ic.ac.uk/info.html
[2]http://hces.bus.olemiss.edu/tools.html

encodings on instance GK01. It is interesting to notice that, as n increases, the gap between the different representations decreases. On a closer look, in spite of some differences, those are minimal. This effect might be explained, to a certain degree, to 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, not present in the binary encoding, is important for tighter instances.

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 others encodings, simply because it has a stronger heuristic bias on its decoder. From column ϱ, it is clear that doesn't happen. The weight-coding scheme only has the best coefficient value for instances PC01 and CB01, and a close second best for instance CB02. For the other two instances, PC02 and GK01, the fitness distance correlation for weight-coding is worse than ordinal and random-key encodings, respectively. The only clear pattern presented for this column is the poorer performance of the binary encoding. In fact, contrary to the previous measure, the fitness distance correlation doesn't 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 a general way, the other four representations have similar values around -0.50, for example, in instances P01, CB01 and CB02, the coefficient is always above -0.50. The clear exception is instance GK01. For the larger instance, the ϱ values are all above -0.20, indicating that for this particular instance, all representations had several difficulties to reach better solutions.

From this table, it is possible to distinguish differences 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, from the decoder with the profit/pseudo-resource consumption ratios, are less visible. There is an indication that the weight-coding representation performs slightly-better than the others but that might be also dependent on the problem instance. In order to gain more discernement on this matter, fitness distance correlation should be examined in association with the autocorrelation measures.

TABLE II
SUMMARY RESULTS FOR AUTOCORRELATION ON ALL THE MKP INSTANCES FOR THE FIVE REPRESENTATIONS.

Representation	Instance	n	m	l	ξ
BR	P01	28	10	7.12	0.25
OR	P01	28	10	4.61	0.16
PR	P01	28	10	7.64	0.27
RK	P01	28	10	10.97	0.39
WC	P01	28	10	**14.51**	**0.52**
BR	P02	50	5	16.18	0.32
OR	P02	50	5	6.23	0.12
PR	P02	50	5	11.72	0.23
RK	P02	50	5	17.05	0.34
WC	P02	50	5	**19.43**	**0.39**
BR	CB01	100	5	**35.74**	**0.36**
OR	CB01	100	5	4.09	0.04
PR	CB01	100	5	7.04	0.07
RK	CB01	100	5	8.12	0.08
WC	CB01	100	5	10.85	0.11
BR	CB02	100	5	**36.65**	**0.37**
OR	CB02	100	5	4.78	0.05
PR	CB02	100	5	6.89	0.07
RK	CB02	100	5	9.22	0.09
WC	CB02	100	5	10.78	0.11
BR	GK01	100	15	**27.32**	**0.27**
OR	GK01	100	15	3.95	0.04
PR	GK01	100	15	8.37	0.08
RK	GK01	100	15	8.98	0.09
WC	GK01	100	15	9.33	0.09

C. Autocorrelation

To determine the correlation length we performed a series of random walks in order to calculate the autocorrelation function for all tested instances, with a phenotype 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 presented in table II. A brief overview of the results reveals a very interesting pattern: for instances P01 and P02, weight-coding representation achieved the highest correlation value, for instances CB01, CB02 and GK01, binary representation attained the highest values. On the other end, ordinal encoding has the lowest autocorrelation values for all problem instances. Another interesting fact is the correlation for all decoder-based representations on instances with 100 items: the values are similar and very low, when comparing to the direct encoding. In fact, binary representation shows consistent values across the instances (between 0.25 and 0.37), which doesn't happen for the indirect encodings.

How can these differences be explained? Part of the answer relies on the the type of mutation variation operator. The lower correlation for the ordinal representation is in compliance with previous studies [11]. Locality indicates that small variations in the genotype space, usually originated by mutation, imply small variations in the phenotype space. Strong locality allows a search algorithm to efficiently explore the neighborhood of the current solutions, whilst 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 lower correlation which implies a more difficult search. In contrast, this does not happen, to a certain degree, on the others representations. In these encodings, the flip mutation operator also changes a single gene but the effect on the

TABLE III

SUMMARY RESULTS FOR FITNESS DISTANCE CORRELATION AND AUTOCORRELATION ON ALL THE MKP INSTANCES FOR THE REPRESENTATIONS WITH ADDITIONAL HEURISTICS AND LOCAL IMPROVEMENT.

Representation	Instances	n	m	$\overline{d_{opt}}$	ϱ	l	ξ
BR	P01	28	10	14.03 (0.36)	0.07	7.12	0.25
BR with LI	P01	28	10	14.33 (0.38)	-0.39	9.93	0.35
BR with PRH	P01	28	10	12.99 (0.43)	-0.58	12.46	0.44
BR with PRH + 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 LI	P02	50	5	26.13 (0.53)	-0.21	15.07	0.30
BR with PRH	P02	50	5	23.93 (0.52)	-0.42	19.78	0.40
BR with PRH + 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 LI	CB01	100	5	50.37 (0.68)	-0.25	15.50	0.15
BR with PRH	CB01	100	5	43.67 (0.87)	-0.43	21.83	0.22
BR with PRH + 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 LI	CB02	100	5	49.94 (0.71)	-0.34	15.79	0.16
BR with PRH	CB02	100	5	43.25 (0.83)	-0.47	19.73	0.20
BR with PRH + 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 LI	GK01	100	15	50.33 (0.70)	-0.06	15.64	0.16
BR with PRH	GK01	100	15	49.80 (0.69)	-0.03	14.86	0.15
BR with PRH + 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**

phenotype is not as dramatic as in the ordinal representation (with the exception of permutations). This effect can explain why ordinal representation has the worse 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 this case, weight-coding and binary representations are shown to have better correlated landscapes than the other representations. Taking into consideration the fitness distance correlation analysis described in the previous section, we can conclude that these encodings should provide a good option for solving the MKP, at least, for mutation-based algorithms.

D. The effect of heuristics and local improvement

To complete our analysis, some additional tests were made regarding the binary and the weight-coding representations. For solving the MKP, these encodings are usually chosen with additional heuristics and/or local improvement [11]. In our tests, we will consider improvements on these encodings:

- *Binary representation with local improvement* (BR with LI): the standard encoding will be subject to a local method that repairs infeasible solutions (by removing items until all constraints are satisfied) and inserting items on a feasible solution (as long as it doesn't violates any constraint);
- *Binary representation with profit/pseudo-resource consumption ratios* (BR with PRH): a feasible solution is built from the genotype according to the ratios;
- *Binary representation with profit/pseudo-resource con-*

sumption ratios and local improvement step (BR with PRH + LI);

- *Weight-coding representation with log-normal* (WC with log-normal): the heuristic is applied on the mutation variation operator . The operator follows a log-normal distribution as:

$$w_j = (1+\gamma)^{N(0,1)}, j = 1, \ldots, n. \quad (14)$$

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 mean 0 and standard deviation 1. We set $\gamma = 0.05$ as recommended by [16].

In table III we present, for all five MKP instances, the results of the applied measures for the four types of binary encodings and two types of weight-coding (we include the results for simple BR and WC for comparison).

A brief perusal of the results reveals one important aspect: the weight-coding representation with the log-normal 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 very near -1.0 (instances P01, CB01 and CB02). When comparing these values to the average minimal distance from the best individual to the optimum, we can observe that they are also the lowest. Only for instances CB01 and GK01 this effect is not visible. Close to the WC with log-normal operator is BR with PRH + LI. 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 found to the optimum (instances CB01 and GK01). Regarding autocorrelation, this encoding does not attain the same degree of success like the weight-coding representation (the values are similar to those attained by the simple direct encoding).

This analysis reveals that adding heuristics and/or local improvement methods to a representation is crucial to obtain better results when solving the MKP. Examining the weight-coding representation, using a mutation variation operator that is more sensible to the problem domain helps to improve performance. Since the floats values represent surrogate multipliers, it is clear that the mutation provided by a uniform distribution will cause strong phenotypic changes, whilst a log-normal distribution will handle the genotype in a more subtle way. The log-normal 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 it's 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 to improve the performance when comparing to the simple direct encoding but unable to achieve the values of the ordinal, permutation and random-key representations. These three previous encodings still perform better than a direct encoding even when we compare them to the binary representation with the ratios heuristic (BR with PRH). The exception is instance P01, although the differences are small. It seems these two improvements, when examined separate 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 ratios heuristic and local improvement are combined, the result is very different and the performance is comparable to the weight-coding representation with the log-normal mutation operator. In terms of autocorrelation, the effects are not the same. For all variants of the direct encoding, the autocorrelation values are similar, only with marginal differences.

As we can see from these results, a pattern arises in terms of the representations performance according to their heuristic bias. This is essential true when looking at the fitness distance correlation coefficient. The representations with stronger heuristic bias (weight-coding with log-normal operator and binary representation with ratios heuristic and local improvement) achieve the best results, very close to -1.0. The encodings with a weak heuristic bias do perform well but clearly below the two previous encodings, while the simple binary representation is the one with the poorer performance. The fitness landscape analysis is compliant with previous studies (e.g., [11]).

VI. Conclusions

In this paper we presented an analysis of fitness landscapes for the Multidimensional Knapsack problem. The main objective of this study was to add a contribution to the investigation of the influence of representations in a combinatorial optimization problem.

The analysis of fitness distance correlation and autocorrelation helps to explain differences in performance achieved by different representations. Within a mutation-based evolutionary algorithm, weight-coding with a log-normal mutation operator and binary representation with a profit/pseudo-resource consumption ratios heuristic and local improvement, appear to be the most suitable representations. 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 an variation operator, as well as the use of local improvement methods can significantly alter the performance for the given representation.

For future work, we plan to extend this research to landscapes produced by crossover operators . This is to complement the current study about the role of representations when solving the Multidimensional Knapsack problem.

Acknowledgments

The first author would like to acknowledge grant SFRH/BD/12615/2003 from *Fundação para a Ciência e a Tecnologia* (FCT), Portugal.

REFERENCES

[1] T. Jones, "Evolutionary algorithms, fitness landscapes and search," Ph.D. dissertation, University of New Mexico, Albuquerque, New Mexico, May 1995.

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

[3] S. Martello and P. Toth, *Knapsack problems: algorithms and computer implementations*. New York, NY, USA: John Wiley & Sons, Inc., 1990.

[4] H. Kellerer, U. Pferschy, and D. Pisinger, *Knapsack Problems*. Springer-Verlag, 2004.

[5] G. R. Raidl, "An improved genetic algorithm for the multiconstrained 0-1 knapsack problem," in *Proc. of the 5th IEEE Conference on Evolutionary Computation, 1998 IEEE World Congress on Computational Intelligence*, Anchorage, Alaska, May 1998, pp. 207–211.

[6] Z. Michalewicz and J. Arabas, "Genetic algorithms for the 0/1 knapsack problem," in *ISMIS '94: Proceedings of the 8th International Symposium on Methodologies for Intelligent Systems*. London, UK: Springer-Verlag, 1994, pp. 134–143.

[7] R. Hinterding, "Mapping order-independent genes and the knapsack problem," in *Proc. of the First IEEE Int. Conf. on Evolutionary Computation*. IEEE Press, 1994, pp. 13–17. [Online]. Available: citeseer.ist.psu.edu/117733.html

[8] 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.

[9] J. Gottlieb, "Permutation-based evolutionary algorithms for multidimensional knapsack problems," in *SAC '00: Proceedings of the 2000 ACM symposium on Applied computing*. New York, NY, USA: ACM Press, 2000, pp. 408–414.

[10] ——, "Evolutionary algorithms for constrained optimization problems," Ph.D. dissertation, Technical University of Clausthal, Clausthal, Germany, 1999.

[11] 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," *Evolutionary Computation Journal*, vol. 13, no. 4, pp. 441–475, December 2005.

[12] J. Gottlieb, "On the feasibility problem of penalty-based evolutionary algorithms for knapsack problems," in *Proceedings of the EvoWorkshops on Applications of Evolutionary Computing*. London, UK: Springer-Verlag, 2001, pp. 50–59.

[13] J. Gottlieb and G. R. Raidl, "Characterizing locality in decoder-based eas for the multidimensional knapsack problem," in *AE '99: Selected Papers from the 4th European Conference on Artificial Evolution*. London, UK: Springer-Verlag, 2000, pp. 38–52.

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

[15] R. Hinterding, "Representation, constraint satisfaction and the knapsack problem," in *Proceedings of the Congress of Evolutionary Computation*, P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, and A. Zalzala, Eds., vol. 2. Mayflower Hotel, Washington D.C., USA: IEEE Press, 6-9 July 1999, pp. 1286–1292.

[16] G. Raidl, "Weight-codings in a genetic algorithm for the multiconstraint knapsack problem," in *Proceedings of the Congress of Evolutionary Computation*, P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, and A. Zalzala, Eds., vol. 1. Mayflower Hotel, Washington D.C., USA: IEEE Press, 6-9 July 1999, pp. 596–603.

[17] J. Gottlieb, "Evolutionary Algorithms for Multidimensional Knapsack Problems: the Relevance of the Boundary of the Feasible Region," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'99)*, W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honovar, M. Jakiela, and R. E. Smith, Eds., vol. 1. San Francisco, California: Morgan Kaufmann, July 1999, p. 787.

[18] G. R. Raidl and J. Gottlieb, "On the importance of phenotypic duplicate elimination in decoder-based evolutionary algorithms," in *Late Breaking Papers at the 1999 Genetic and Evolutionary Computation Conference*, S. Brave and A. S. Wu, Eds., Orlando, Florida, USA, 13 July 1999, pp. 204–211.

[19] J. Branke, E. Salihoglu, and S. Uyar, "Towards an analysis of dynamic environments," in *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*. New York, NY, USA: ACM Press, 2005, pp. 1433–1440.

[20] S. Wright, "The roles of mutation, inbreeding, crossbreeding and selection in evolution," in *Proceedings of the VI International Conference on Genetics, Vol. 1*, 1932, pp. 356–366.

[21] P. Merz, "Memetic algorithms for combinatorial optimization problems: Fitness landscapes and eective search strategies," Ph.D. dissertation, Department of Electrical Engineering and Computer Science, University of Siegen, Germany, 2000. [Online]. Available: citeseer.ist.psu.edu/article/merz01memetic.html

[22] GLPK, "Gnu linear programming kit. http://www.gnu.org/software/glpk/glpk.html."