The TransRAR Crossover Operator for Genetic Algorithms with Set Encoding

Ruben Ruiz-Torrubiano
Computer Science Department
Universidad Autonoma de Madrid
Madrid, Spain
ruben.ruizt@estudiante.uam.es

Alberto Suarez
Computer Science Department
Universidad Autonoma de Madrid
Madrid, Spain
alberto.suarez@uam.es

ABSTRACT

This work introduces a new crossover operator specially designed to be used in genetic algorithms (GAs) that encode candidate solutions as sets of fixed cardinality. The Transmitting Random Assortment Recombination (TransRAR) operator proceeds by taking elements from a multiset, which is built by the union of the parent chromosomes, allowing repeated elements. If an element that is present in both parents is drawn, it is accepted with probability 1. Elements that belong to only one of the parents are accepted with a probability p, smaller than 1. The performance of this novel crossover operator is assessed in synthetic and real-world problems. In these problems, GAs that employ this type of crossover outperform those that use alternative operators for sets, such as Random Assortment Recombination (RAR), Random Respectful Recombination (R^3) or Random Transmitting Recombination (RTR). Furthermore, TransRAR can be implemented very efficiently and is faster than RAR, its closest competitor in terms of overall performance.

Categories and Subject Descriptors

F.2 [**Analysis of Algorithms and Problem Complexity**]: Miscellaneous

General Terms

Algorithms

Keywords

Crossover operators, genetic algorithms, forma theory

1. INTRODUCTION

In most practical applications for genetic algorithms (GA), the candidate solutions are encoded as strings of fixed length over a given alphabet. A common choice is a binary alphabet [3]. In binary encoding, a value "1" in the $i-$th position of the string (*locus*) can be used to represent the presence of a given property and the value "0" its absence. Standard evolutionary operators used in combination with this encoding are the one-point, the two-point or the uniform crossover operators. This type of encoding is unnatural when dealing with optimization problems with cardinality constraints [2, 5, 19]. These constraints limit the maximum number of elements that can be included in the final solution. Cardinality-constrained knapsack problems [6], optimal portfolio selection [11] or the tracking of financial indexes [21] are practical examples of such problems. In these problems, using a binary representation and standard crossover operators can lead to violations of the cardinality constraints. Consider for instance a problem whose solution requires the selection of 2 out of 5 elements (e.g. selecting which stocks to include in an investment portfolio). Let the strings 11000 (select the first and the second element) and 10010 (select the first and the fourth element) be two chromosomes representing two candidate solutions. If one-point crossover is used, and the crossover point selected is 2, one of the resulting children is 11010 (select the first, the second and the fourth element), whose cardinality is different than the parents'. The offspring that violate the cardinality constraint must be either penalized by using a suitably tuned penalty function, or repaired by special procedures. Both approaches tend to misguide the search process [14].

Alternatively, candidate solutions can be encoded as sets of fixed cardinality. Using appropriately designed evolutionary operators that preserve the cardinality of the candidate solutions, it is possible to avoid the generation of infeasible individuals in the course of the evolution. For instance, one can use a mutation operator that consists in exchanging an element in the set with an element outside the set. This operator preserves the cardinality of the solution and therefore produces only feasible individuals. As crossover operator we can take the intersection of the parents' sets and add elements present in only one parent randomly until the desired cardinality is reached. In [16] a general framework for exploiting problem-specific information in the design of genetic operators and representations is presented. This framework, known as *forma theory*, is a generalization of schema theory. Based on an analysis of the properties that recombination operators should have, a number of representation-independent crossover operators are defined. Among these are the Random Assortment Recombination (RAR), the Random Respectful Recombination (R^3) and the Random Transmitting Recombination (RTR) operators. One can readily design special versions of these operators that preserve the cardinality of the parents in the offspring.

GECCO'11, July 12–16, 2011, Dublin, Ireland.

In this work, we introduce the Transmitting Random Assortment Recombination (TransRAR) operator. This operator implements recombination using a principled approach, analogous to the one employed by Radcliffe in the design of RAR. One of the main differences between TransRAR and RAR is that, whereas RAR tries to balance *respect* and *assortment*, TransRAR considers a stronger notion than respect called *transmission*. Respect requires that the children generated by the recombination process include all the elements that present in both parents (intersection set). Transmission requires that every element in the child chromosome comes from at least one of the parents (union set). In RAR the amount of respect achieved can be tuned by modifying a parameter w: the larger the value of w, the higher degree of respect. By contrast, TransRAR attempts to strike a balance between transmission and assortment by assigning a higher probability of being selected to those elements that are present in both parents. Elements that are present in only one parent are accepted only with probability p.

The manuscript is organized as follows: Section 2 provides an overview of existing crossover operators for sets. Section 3 describes the TransRAR operator and analyzes the complexity of its implementation. Section 4 analyzes the performance of GAs that employ this operator in benchmark synthetic and real-world problems. Section 5 summarizes the results and conclusions of this work and outlines directions for future research.

2. CROSSOVER OPERATORS FOR SETS

In this section we review some crossover operators for sets, whose properties and performance have been investigated in the literature. Most of these operators were introduced by [17] using ideas from *forma theory*. Forma theory is a generalization of schema theory, that uses *formae* instead of schemata as the basis for analysis. A forma is an equivalence class (a disjoint partition of the search space) resulting from the definition of equivalence relations between candidate solutions for the problem at hand. For instance, let x and y be two candidate solutions for a given problem. We define the equivalence relation R_1 as xR_1y if and only if x and y contain element 1. This divides the search space into two disjoint partitions: those solutions which contain element 1, and those which do not. In general, a set of m equivalence relations $\{R_1, \ldots, R_m\}$ must be constructed in order to properly define a genetic representation. For instance, if our problem consists in finding an optimal subset of a set of N elements, we need N equivalence relations $\{R_1, \ldots, R_N\}$ where xR_iy if and only if x and y contain element i. Thus, it is necessary to use problem-specific knowledge in the definition of these equivalence relations.

The motivation of forma theory is the observation that if schemata can not group solutions with similar performance, the predictions of the Schema Theorem [4] about the performance of the solutions in the next generation will fail. Therefore, the algorithm will not be able to perform better than an enumerative search. We expect that by defining crossover operators that preserve formae rather than schemata, the search can be guided effectively and obtain improved results.

In forma theory, an equivalence relation plays the role of a gene. The induced equivalence classes are therefore analogue to the alleles of the gene. For instance, an equivalence relation for eye color induces a partition of the search space of all individuals where the disjoint partitions (alleles) are blue, green and brown [17]. Recombination operators should then manipulate these equivalence relations in a meaningful way. In order to define what "meaningful" is in this context, some desirable properties are introduced:

- *Respect* refers to the characteristic that the children produced by a recombination operator should always include all alleles present in both parents. For instance, if the recombination operator $\mathcal{K}$ is respectful, and two individuals represented by sets $\{1, 2, 3\}$ and $\{1, 4, 5\}$ are recombined, then all children generated by $\mathcal{K}$ will contain element 1.

- *Assortment* requires that every combination of the alleles of the parents is possible in the offspring. For instance, if $\mathcal{K}$ has this property, then from parents $\{1, 2, 3\}$ and $\{1, 4, 5\}$ the child $\{2, 3, 5\}$ can be produced by $\mathcal{K}$ (and also other combinations).

- *Transmission* is a stronger notion than respect. It requires that every allele in the children should come from at least one of its parents. Clearly, if $\mathcal{K}$ is a respectful recombination operator, then $\mathcal{K}$ also transmits genes, because alleles common to both parents are transmitted to the children.

In some genetic representations, operators that satisfy both respect and assortment cannot be built. We say in this case that the formae are *non-separable*. This is the case for sets of specified cardinality, where basic formae are defined as the belonging of each possible element in the set. For instance, consider the case where we recombine sets $\{1, 2, 3\}$ and $\{1, 4, 5\}$ and each solution must exactly have 3 elements. Respect requires element 1 be present in all children, but assortment implies that the set $\{2, 3, 5\}$ must be a possible outcome. Therefore, no operator can exist which both respects and assorts these formae.

Let n be the required number of elements in the final solution of a cardinality-constrained problem. Let N be the number of all available elements ($n < N$). We now describe some possible operators on sets of fixed cardinality n that are designed to take into account the properties respect, assortment or transmission [16].

- *Random Respectful Recombination* R^3
  The R^3 operator is defined as follows: Take two sets A and B and calculate their intersection $A \cap B$. The child is initialized as $C = A \cap B$. If $|C| = n$ the recombination step finishes. Otherwise, the child is completed with elements chosen at random from the rest of elements present in the parents. This operator clearly respects all formae to which both parents belong.

- *Random Transmitting Recombination RTR*
  This operator chooses randomly an element of the set of all sets of cardinality n that can be constructed with all elements present in the parent sets. That is, take the union of the two parents $A \cup B$ and choose elements from this set until the child is complete. RTR clearly enforces gene transmission.

- *Random Assortment Recombination RAR*
  This operator (defined in Algorithm 2.1) is designed for problems with non-separable formae, in which respect and assortment cannot be simultaneously achieved. The integer parameter w regulates how much common information from the parents is to be exploited by the

operator. In the pseudocode in Alg. 2.1, A is the intersection set, which contains elements that appear in both parents. Set B includes the elements not present in any parent. Sets C and D are identical and contain the elements present in only one parent. Set E is initially empty ($E = \emptyset$). An additional set G is then built using w copies of the elements from A and B and one copy from the elements in C and D. The elements in G are labeled according to the set from which they originate. A child chromosome is generated by extracting one element from G in each iteration. Let g be the element which is extracted from G: if it originally comes from A or C and $g \notin E$, then it is included in the genome of the child. If $g \in B$ or $g \in D$, then it is included in set E. The process is terminated when n elements have been included in the child, or when $G = \emptyset$. If the latter occurs, then the child is completed with elements selected at random from those that have not been included in the child. Clearly, this procedure assorts formae. The larger the value of w, the more respectful the operator is. Asymptotically, RAR approaches R^3 as $w \to \infty$.

Algorithm 2.1 Random Assortment Recombination algorithm

1. Create auxiliary sets A, B, C, D, E:
   - A = elements present in both parents.
   - B = elements not present in any of the parents.
   - $C \equiv D$ elements present in only one parent.
   - $E = \emptyset$.
2. Build set G with w copies of elements from A and B, and 1 copy of the elements in C and D.
3. Initialize child chromosome $\phi = \emptyset$.
4. While $|\phi| < n$ and $G \neq \emptyset$:
   - Extract $g \in G$ without replacement.
   - If $g \in A$ or $g \in C$, and $g \notin E$, $\phi = \phi \cup \{g\}$.
   - If $g \in B$ or $g \in D$, $E = E \cup \{g\}$.
5. If $|\phi| < n$, add elements not yet included chosen at random until chromosome is complete.

GAs with set encoding have been extensively investigated in the literature and applied in a variety of contexts: The authors of [2] found that the performance of RAR strongly depends on the ratio n/N (where n is the size of the subset and is N the size of the element universe). They showed that the diversity in the population is maximal for $n/N = 0.5$ and very small for values close to 0 or 1. The consequence is premature convergence, specially in cases where $n/N\,(1 - n/N)$ is small. To avoid this problem they proposed some improvements in the design of the RAR operator. Specifically, the diversity in the population is enhanced when it decreases in the course of the evolutionary process. The recombination operators designed depend on additional parameters. Determining the optimal values of these parameters can be difficult and represents a handicap for the practical implementation of these operators.

Variants of the RAR and the R^3 crossover operators [16] are applied in [10] to the p-Median problem. In these modifications, local search is used to improve the fitness of the offspring obtained in the recombination operation. In [5], the R^3 operator was found to give better results than a greedy algorithm and than a GA with a special string encoding in the leaf-constrained minimum spanning tree problem. The goal of this problem is to build a spanning tree on a undirected weighted graph such that it contains more than l leaves and whose weight is as small as possible. As the authors pointed out, the use of a fixed-length subset encoding and operators that always produce feasible solutions improves the efficiency of the algorithm because it reduces the size of the search space. In [19] the performance of GAs that use a binary representation with penalty or repair mechanisms and of GAs that use a set representation and implement recombination using the RAR operator are compared in a series of benchmark problems with cardinality constraints. A hybrid approach using the RAR crossover operator obtained the best overall results in the evaluation.

3. TRANSMITTING RAR

In this section we introduce the Transmitting RAR (TransRAR) crossover operator. The key idea in the design of this operator is to guarantee gene transmission. By contrast, the principle used in the design of the RAR operator is to achieve an appropriate balance between respect and assortment. Transmission is preferable than strict respect because it favors genetic diversity while guaranteeing that the genetic material of the parents' chromosomes will be transmitted to their offspring.

The pseudocode in Algorithm 3.1 describes the implementation of the TransRAR crossover operator. The operator assorts formae because every combination of alleles in the parents can be obtained with a certain probability. It also transmits genes: if they are selected, alleles that are present in both parents are always accepted. Alleles that are present in only one of the parents are accepted with probability p. The value of p controls the balance between respect and transmission. For lower values of p the operator favors respect. In the limit $p = 0$ all elements present in only one parent are rejected and the maximum level of respect is achieved. Note that for this value, the probability that U is empty before the child is completed is 1 if the parents are not equal. For higher values of the parameter p, more elements present in only one parent are selected on average. If the degree of respect is too high (p too low), the genetic algorithm tends to convergence prematurely and get trapped in local optima. The larger the magnitude of p, the higher the variability in the offspring population produced by the operator. As a consequence, the evolutionary process can become less effective. An adequate balance between respect and diversity is achieved for intermediate values of p. In this work, the value $p = 1/2$ is used on the basis of exploratory experiments. It provides good overall results in all the problems investigated.

3.1 Complexity analysis

The TransRAR operator can be implemented very efficiently. In particular, the following upper-bound can be given for the time-complexity of the algorithm: Let n be the size of the parents' chromosomes and N the size of the universe from which elements can be selected. Let Φ_1 and Φ_2 be the parent chromosomes of cardinality n, and let $\mathcal{N} = \Phi_1 \cup \Phi_2$. Let the *extension function* for the parent

Algorithm 3.1 The TransRAR crossover operator for sets.

INPUT: Φ_1, Φ_2 the parent chromosomes of cardinality n.
OUTPUT: Φ offspring of cardinality n.

1. Create multiset U as the multiset-union of the parent chromosomes: $U = \Phi_1 \uplus \Phi_2$.
2. Assign each element $u \in U$ the attribute $E_{\Phi_1\Phi_2}(u)$.
3. While child chromosome ϕ is incomplete ($|\phi| < n$):
   - Extract an element u_k from U uniform randomly. $U = U \setminus \{u_k\}$
   - If $E_{\Phi_1\Phi_2}(u_k) = 1$, then $\phi = \phi \cup \{u_k\}$ with probability 1.
   - else, $\phi = \phi \cup \{u_k\}$ with probability p.
   - If $U = \emptyset$, select $n - |\phi|$ elements randomly to complete chromosome.

chromosomes Φ_1 and Φ_2 be defined as

$$\begin{aligned} E_{\Phi_1\Phi_2} &: \mathcal{N} \to \{0,1\} \\ E_{\Phi_1\Phi_2}(u) &= \begin{cases} 1 & \text{if } u \in \Phi_1 \cap \Phi_2 \\ 0 & \text{if } u \in \Phi_1 \cup \Phi_2 - (\Phi_1 \cap \Phi_2) \end{cases} \end{aligned} \quad (1)$$

Steps 1 and 3 require $O(n)$ operations. Consider Step 2. One possible method to calculate the function $E_{\Phi_1\Phi_2}$ is to sort the elements in U and then remove one element u for which we calculate $E(u)$. Note that element u is repeated if and only if it is in the intersection set of the two parent sets. Then we can apply binary search to look for the presence of an additional copy of u. If an element is found, then $E_{\Phi_1\Phi_2}(u) = 1$. Otherwise $E_{\Phi_1\Phi_2}(u) = 0$. Since sorting requires $O(n \log n)$ steps and removing and searching in the sorted multiset require $O(\log n)$ steps, the worst case complexity $f(n)$ of the algorithm is

$$f(n) = O(n) + O(n \log n) = O(n \log n) \quad (2)$$

The time-complexity of the RAR operator is computed for a fixed value of the parameter $w > 0$: Let n and N be defined as before. Step 1 in 2.1 can be completed in $O(N) + O(n \log n)$ operations, assuming that we have to sort the parent chromosomes first. Suppose that building the multiset G requires constant time $O(1)$. Step 4 requires in the worst case exactly $|G|$ iterations. By construction, $|G| = O(wN)$. Determining from which set element g comes requires constant time if each element is labeled when constructing G. Assuming that set E is kept sorted on every step, then searching g in E requires $O(\log N)$ steps. Therefore, the total worst-case complexity is

$$\begin{aligned} f(N,n,w) &= O(1) + O(N) + O(n \log n) + O(wN \log N) = \\ &= O(wN \log N) \end{aligned}$$

Note that the worst-case complexity of TransRAR is expressed in terms of the size of the subset n, whereas in the case of RAR the complexity if a function of the size of the *total* number of elements, $N > n$. In many cases of practical interest $N >> n$. Therefore, the worst-case complexity of RAR is larger than TransRAR. This does not necessarily imply that TransRAR is more efficient than RAR in average cases. Nonetheless, the empirical evidence presented in the following section shows that TransRAR is much more efficient than RAR in the problems investigated.

4. EMPIRICAL EVALUATION

In this section we present the results of experiments carried out to assess the performance of the TransRAR operator in several synthetic (knapsack problem, epistatic functions) and real-world (portfolio selection in finance) problems of practical interest.

4.1 The knapsack problem

Consider the problem of filling a knapsack with D items. Associated with each element i is a profit p_i and a weight w_i. The knapsack has a limit W on the total weight it can carry. The objective is to find the subset of elements that maximize the total profit without exceeding the maximum weight W. More formally, we seek to find the optimal solution to the following 0/1 integer linear problem (ILP):

$$\max \quad \sum_{i=1}^{D} p_i z_i \quad \text{s.t.} \sum_{i=1}^{D} w_i z_i \leq W \quad z_i \in \{0,1\} \quad (3)$$

where $z_i = 1$ if element i is included in the knapsack, and 0 otherwise. In this formulation, there are no cardinality constraints. Nonetheless, the optimal solution to the unconstrained problem can be obtained by solving D knapsack problems with cardinality constraints

$$\sum_{i=1}^{D} z_i = k; \; k = 1, 2, \ldots D \quad (4)$$

The 0/1 knapsack problem has been approached using both exact and approximate methods. Exact algorithms based on branch-and-bound techniques and dynamic programming [12] are reviewed in [15]. Approximate methods, such as genetic algorithms [22, 7], estimation of distribution algorithms [8] and ant colony optimization [9] have also been used to address this problem. We present results on the knapsack problem using the testing protocol proposed in [13] [22]. Let $v, r \in \mathbb{R}^+$, $v > 1$. In terms of these parameters, the following types of knapsack problems can be defined:

(1) *Uncorrelated*: The values w_i and p_i are obtained in independent samples from a uniform distribution in the interval $[1, v]$.

(2) *Weakly correlated*: The weights w_i are uniformly distributed in $[1, v]$ and the profits p_i in $[w_i - r, w_i + r]$.

(3) *Strongly correlated*: The weights w_i are uniformly distributed in $[1, v]$, and the profits are a deterministic function of the weights $p_i = w_i + r$.

Correlated problems (both weakly and strongly correlated) are often more difficult to solve than uncorrelated problems. Following the protocol employed in [19], we use $v = 10$, $r = 5$ and a capacity $W = 2v$, which leads to solutions in which only a few items are selected. The size of the problems ranges from 100 to 1000 elements in the knapsack. The results reported are averages over 25 independent random realizations of each problem. The population size in the GA is set to 100 individuals. The probabilities of crossover and mutation are $p_c = 1$, $p_m = 10^{-2}$, respectively.

The value of the parameter p in the TransRAR operator (Alg. 3.1) is determined in exploratory experiments. Figure 1 presents a typical outcome of these experiments. This

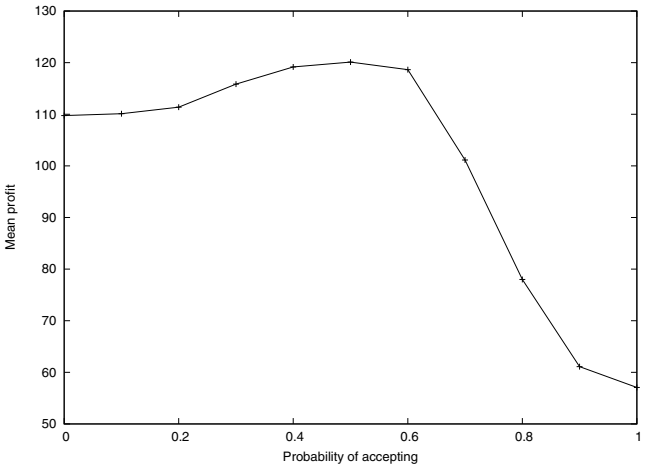

Figure 1: Mean profit obtained by TransRAR in the knapsack problem with 500 items and no correlation as a function of the probability of acceptance.

figure displays the best average profit obtained in a knapsack problem with 500 elements and no correlation as a function of p, the probability of accepting an element that is present in only one of the parents once it has been drawn. As can be seen from this plot, large values of p lead to a sharp deterioration of performance. The reason is that too much variability is introduced in the search and the algorithm is not able to preserve formae that perform well. Values of p close to 1/2 yield the best results. Fairly good results are also obtained for lower values of p. In this particular case, high degrees of respect (and therefore lower variability) lead to good results. This is true in general, but too much respect can result in premature convergence and therefore one typically obtains suboptimal solutions. Experiments in other problems give similar results. Therefore, no further adjustments of this parameters are made and the value $p = 1/2$ is used in all cases. Exploratory experiments are carried out to determine the optimal value of the parameter w in RAR. In view of the results of these experiments, the best performance is obtained with $w = 1$. This is different from the value $w = 2$ proposed as a natural choice and used in the original study by Radcliffe [18]. In our experiments, this value produced suboptimal solutions because too much common information from the parents is exploited by the crossover operator. The consequence is that the algorithm frequently converges prematurely and becomes trapped in local optima.

The results of the experiments in knapsack problems are summarized in Table 1. The performance of each method on the different problems is measured in terms of average profit obtained and average time (in seconds) required to reach a solution. All experiments are carried out on an Intel Core Duo machine with 2 GHz clock speed and 2 GB RAM. In most cases the best results are obtained by TransRAR. Nevertheless, the differences in quality between the solutions obtained by GA-RAR and by GA-TransRAR are very small. In terms of efficiency, the computational cost of TransRAR is much lower than RAR, RTR and R^3. For instance, TransRAR is 6.6 times faster that RAR in the weak correlation problems with 1000 elements. The smallest speed-up ratio between these two methods is 2.1 in the strong correlation case with a universe of 100 elements.

4.2 Epistatic functions

Epistatic functions were introduced in [18] as a benchmark to evaluate and compare the performance of set-based crossover operators. These functions are constructed as follows: Given a universe size N, the objective is to find the subset $s^* = \{1, 2, \ldots, n\}$ where $n = N/2$. In a non-epistatic problem, the credit of a candidate solution s is the number of elements it has in common with the optimal solution s^*. Therefore, the fitness function is $F(s) = |s \cap s^*|$.

In an epistatic (or deceptive) version of problem, the credit for having k elements in common with the optimal solution is randomly shuffled. This is done as follows: First, construct consecutive bins of a fixed size L. In the first bin we put the elements from 1 to L. In the second bin, the elements from $L+1$ to $2L$, and so on. Therefore, each bin i contains the elements from $iL+1$ to $(i+1)L$, for $i = 0, \ldots, N/L-1$. Candidate solutions receive full credit for a particular bin only if they include all the elements in the bin. This means that, for bin i, solution s obtains L points if it contains all elements from $iL+1$ to $(i+1)L$. The credit for having fewer elements is a randomly chosen value in the interval $[0, L-1]$. For instance, in a problem whose selection universe contains $N = 120$ items, $L = 5$, we construct 24 bins. The first bin contains the elements from 1 to 5, the next from 6 to 10, and so on. The credit for having *all* elements from 1 to 5 is exactly 5, but the credit for having the elements $\{1, 2, 3, 4\}$ is randomly chosen in the interval $[0, 4]$. Note that a candidate solution that includes items $\{1, 2, 3\}$ can therefore receive higher credit than another one including $\{1, 2, 3, 4\}$. Clearly, the larger the bin size L, the more deceptive and the more difficult the problem is. Therefore, the bin size is used as a measure of the epistasis of the problem.

The results for $N = 120$ and different degrees of epistasis (bin size) are presented in Table 2. Analyzing these results one concludes that both RAR and TransRAR achieve the optimal solution in the non-epistatic case. For lower degrees of epistasis, TransRAR obtains the best results. Nonetheless, in the problem with the highest degree of epistasis (bin size 15) the best results are obtained by RAR, followed by R^3. Interestingly, the quality of the solutions obtained by RTR and R^3 increase with the degree of epistasis. This is because the problem becomes more random as the degree of epistasis increases. Therefore all algorithms tend to perform equally well or equally poorly when the problem becomes random. Note that again, TransRAR is in all instances more efficient than RAR. In this case, the speed-up factors range from 1.7 to 1.8.

4.3 Portfolio selection with cardinality constraints

The problem of portfolio selection [11] with cardinality constraints can be summarized as follows: Let $w_i \in [0, 1]$ represent the proportion of wealth invested in product i in a given portfolio. The goal is to construct a portfolio $P = \{w_1, w_2, \ldots, w_N\}$ such that it minimizes a measure of risk (in the Markowitz model, risk is measured in terms of the variance of the portfolio returns) for a expected value of the portfolio returns. The portfolio invests at most in $n \leq N$ assets and satisfies certain constraints of practical importance. The precise formulation of the resulting mixed-

Table 1: Results for the 0-1 Knapsack problem with restrictive capacity

Corr.	No.Items	Algorithm							
		GA RTR		GA R^3		GA RAR		GA TransRAR	
		Profit	Time	Profit	Time	Profit	Time	Profit	Time
none	100	80.8419	34.8	80.7396	24.6	**82.0870**	46.0	81.9757	20.9
none	250	99.8103	53.5	96.6147	34.7	105.3458	106.3	**105.4573**	31.9
none	500	109.4330	66.1	104.6797	42.4	119.8828	199.9	**120.1059**	41.9
none	750	112.6248	76.9	107.2820	46.0	**126.2109**	286.3	126.0044	50.0
none	1000	113.2639	88.0	108.2204	51.6	128.9135	373.9	**128.9246**	56.8
weak	100	53.3890	34.4	52.9563	24.2	54.3782	45.6	**54.3814**	21.0
weak	250	63.6689	51.2	61.8034	32.6	66.2444	102.6	**66.6882**	30.8
weak	500	68.4328	66.8	65.2095	41.1	74.1670	200.4	**74.7777**	42.7
weak	750	70.2186	78.2	68.3203	47.4	77.2306	289.0	**78.3826**	50.0
weak	1000	70.2327	94.8	67.5378	50.7	**80.3405**	375.6	79.9027	57.1
strong	100	78.9795	37.1	78.7587	25.8	79.7744	48.4	**79.7759**	22.5
strong	250	92.3969	57.3	89.9881	36.5	94.1974	109.0	**94.1989**	33.3
strong	500	96.1989	70.8	94.7948	49.8	101.3986	204.0	**101.3987**	43.2
strong	750	98.7989	83.0	96.1836	44.6	**104.7966**	300.9	104.7946	51.6
strong	1000	98.3991	90.5	97.1871	54.4	**106.5970**	388.0	106.1977	59.1

Table 2: Results for the epistatic functions

Problem type	Algorithm							
	GA RTR		GA R^3		GA RAR		GA TransRAR	
	Fitness	Time	Fitness	Time	Fitness	Time	Fitness	Time
none	53.44	25.7	59.04	16.6	**60.00**	16.7	**60.00**	9.8
Bin=5	40.12	26.9	47.16	17.9	48.72	17.6	**49.00**	10.4
Bin=10	44.40	27.1	46.48	17.9	46.44	18.5	**47.00**	10.2
Bin=12	47.04	27.2	47.88	18.2	47.88	18.8	**47.96**	10.3
Bin=15	49.12	27.2	48.96	18.4	**49.36**	18.9	48.92	10.3

integer quadratic program (MIQP) is

$$\min_{\mathbf{w},\mathbf{z}} \quad \mathbf{w}^{[\mathbf{z}]^{\mathrm{T}}} \cdot \boldsymbol{\Sigma}^{[\mathbf{z},\mathbf{z}]} \cdot \mathbf{w}^{[\mathbf{z}]} \tag{5}$$

$$\text{s.t.} \quad \mathbf{w}^{[\mathbf{z}]^{\mathrm{T}}} \cdot \bar{\mathbf{r}}^{[\mathbf{z}]} = R^* \tag{6}$$

$$\mathbf{w}^{\mathrm{T}} \cdot \mathbf{1} = 1, \;\; \mathbf{w} \geq \mathbf{0} \tag{7}$$

$$\mathbf{l}^{[\mathbf{z}]} \leq \mathbf{w}^{[\mathbf{z}]} \leq \mathbf{u}^{[\mathbf{z}]}, \;\; \mathbf{l}^{[\mathbf{z}]} \geq \mathbf{0}, \;\; \mathbf{u}^{[\mathbf{z}]} \geq \mathbf{0} \tag{8}$$

$$\mathbf{z}^{\mathrm{T}} \cdot \mathbf{1} \leq n \tag{9}$$

The vector $\mathbf{z}^t = \{z_1, \ldots, z_N\}$ is composed of binary variables z_i that indicate whether product i is included in the portfolio ($z_i = 1$) or not ($z_i = 0$). We denote $\mathbf{x}^{[\mathbf{z}]}$ as the reduced vector obtained by removing from $\mathbf{x}$ those components for which $z_i = 0$. Analogously, the matrix $\boldsymbol{\Sigma}^{[\mathbf{z},\mathbf{z}]}$ denotes the reduced matrix obtained by removing those rows and columns for which the corresponding binary variable is zero ($z_i = 0$). This model requires as input $\bar{\mathbf{r}}$, the vector of expected asset returns, and $\boldsymbol{\Sigma}$, the covariance matrix for the portfolio returns. Additionally, the value R^* is specified by the investor. It is the level of expected return of the portfolio, as specified by the linear constraint (6). Different portfolios are obtained as solutions of Eqs. (5)-(9) by selecting different values of R^* in the interval $[R^*_{min}, R^*_{max}]$, where $R^*_{min}(R^*_{max})$ is the lowest (highest) expected return of the assets that are considered for investment. This set of Pareto-optimal portfolios form the *efficient frontier*. Each point in this frontier corresponds to the portfolio that minimizes the risk for the specified level of expected return. In a dual view of the problem, it also corresponds to a portfolio that maximizes the expected return for a given level of risk.

Equation 7 is a budget constraint. It is written in terms of the vector of investment weights and a vector of ones $\mathbf{1}$. The constraint reflects the fact that the initial wealth is invested fully in the portfolio (i.e. no transaction costs are considered). This constraint also specifies that short-selling is not allowed ($w_i \geq 0$) and that one cannot borrow money to invest in risky assets ($w_i \leq 1$). Eq. (8) reflects the fact that the investor can set lower and upper bounds on the weights of each of the assets selected for investment ($l_i \leq w_i \leq u_i$). Finally, Eq. (9) is the cardinality constraint, which limits the number of assets included in the final portfolio. This constraint can be introduced to avoid investing in too many products, which may increase the difficulty of managing the portfolio.

Experiments are carried out with the values $l_i = 0.1$, $u_i = 1.0, i = 1, \ldots, N$ and $n = 10$ as in previous studies [20] [14]. For each investment problem (defined by a different universe of assets for investment) we compute the optimal risk of $N_F = 100$ portfolios obtained by fixing a different value of the expected return R^*. These vales are equally spaced in the interval $[R^*_{min}, R^*_{max}]$. The quality of the solutions is determined by calculating the average relative distance between the constrained efficient frontier (σ^c_i, risk of the optimal portfolio with cardinality constraints) and the unconstrained efficient frontier (σ^*_i, optimal risk without cardinality constraints)

$$D = \frac{1}{N_F} \sum_{i=1}^{N_F} \frac{\sigma^c_i - \sigma^*_i}{\sigma^*_i} \tag{10}$$

The problems compiled in the OR-Library [1] are used in the

experiments. The goal is to construct optimal portfolios that invest in the universe of assets included in the calculation of different stock market indexes: Hang Seng (Hong-Kong, 31 assets), DAX (Germany, 85 assets), FTSE (UK, 89 assets), Standard and Poor's (U.S.A., 98 assets) and Nikkei (Japan, 225 assets). In the genetic algorithm populations of 100 individuals are used. Mutation and crossover probabilities are set to $p_m = 10^{-2}$ and $p_c = 1$.

The results of the experiments performed are summarized in Table 3. The value of D displayed in the third column is the best out of 5 executions of the algorithms for each instance. The column labeled as *success rate* gives the proportion of attempts in which the best known solution is reached. The last two columns report the time employed (in seconds) and the number of quadratic optimizations performed, respectively. The table shows that RAR and TransRAR obtain the best results. TransRAR is also more efficient than RAR in all problem instances. The speed-up factors range between 1.1 in the Hang Seng problem and 1.6 in the Nikkei problem. Moreover, the success rates are always the largest for TransRAR. Note that the number of quadratic optimizations is similar both in TransRAR and RAR, which indicates that the reason for the efficiency of the algorithm is the implementation of the crossover operator itself and not the fact that fewer quadratic optimizations are performed.

5. CONCLUSIONS

In this work, we have introduced a novel crossover operator for problems whose candidate solutions are encoded as sets of fixed cardinality. The TransRAR operator is designed on the basis of ideas from forma theory. Besides preserving the cardinality of the candidate solutions, the TransRAR operator enforces transmission. In TransRAR crossover, elements that are present in both parents are accepted whenever they are drawn. By contrast, elements that are present in only one parent are accepted only with a probability $p < 1$ upon selection. The transmission property ensures that the genetic material from the parents is included in the children. At the same time it allows to control the variability produced by the operator by selecting the value of p. Transmission is a more restrictive property than respect, in the sense that all operators that enforce respect also enforce transmission. However, not all operators that enforce transmission are respectful. The amount of variability produced by respectful crossover operators is often insufficient to guarantee an effective exploration of the state space, which means that the search often gets trapped in local optima.

The operations required to perform TransRAR recombination can be implemented very efficiently. The worst-case performance of the algorithm is $O(n \log n)$, where n is cardinality of the set. By contrast, the time complexity of the implementation of the RAR operator is $O(wN \log N)$, where N is the total number of available elements and w measures the degree of common information from the parents which is exploited in the RAR operator. Experiments in a suite of benchmark problems (knapsack, epistatic functions, portfolio selection with cardinality constraints) show that GAs with TransRAR outperform GAs with RAR and have lower computational costs.

Directions of future research include the analysis of the performance of the TransRAR crossover operator with the help of forma theory. In particular, the role of the parameter p (probability of acceptance of elements included only in one of the parents) will be investigated within this framework.

6. ACKNOWLEDGMENTS

This research has been supported by the Spanish Dirección General de Investigación, project TIN2010-21575-C02-02.

7. REFERENCES

[1] J. E. Beasley. OR-library: Distributing test problems by electronic mail. *Journal of the Operational Research Society*, 41 (11):1069–1072, 1990.

[2] K. D. Crawford, C. J. Hoelting, R. L. Wainwright, and D. Schoenefeld. A study of fixed-length subset recombination. In *R. K. Belew and M. D. Vose (Eds.) Foundations of Genetic Algorithms 4*, pages 365–378. Morgan Kaufmann, 1997.

[3] D. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Weasley, Reading, MA, 1989.

[4] J. H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[5] B. A. Julstrom. Codings and operators in two genetic algorithms for the leaf-constrained minimum spanning tree problem. *International Journal of Applied Mathematics and Computer Science*, 14(3):385–396, 2004.

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

[7] S. Ku and B. Lee. A set-oriented genetic algorithm and the knapsack problem. In *Proceedings of the IEEE World Congress on Evolutionary Computation (CEC2001)*, 2001.

[8] P. Larrañaga and J. A. Lozano, editors. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2002.

[9] C. Lee, Z. Lee, and S. Su. A new approach for solving 0/1 knapsack problem. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics 2006, October 8-11, Taipei, Taiwan*, 2006.

[10] A. Lim and Z. Xu. A fixed-length subset genetic algorithm for the p-median problem. In *Genetic and Evolutionary Computation GECCO 2003, Lecture Notes in Computer Science*, volume 2724, pages 212–213, 2003.

[11] H. Markowitz. Portfolio selection. *Journal of Finance*, 7:77–91, 1952.

[12] X. Meng, Y. Zhu, and X. Wu. Improved dynamic programming algorithms for the 0-1 knapsack problem. In *Proceedings of the 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT)*, 2010.

[13] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer Verlag, 1996.

[14] R. Moral-Escudero, R. Ruiz-Torrubiano, and A. Suárez. Selection of optimal investment portfolios with cardinality constraints. In *Proceedings of the IEEE World Congress on Evolutionary Computation*, pages 2382–2388, 2006.

[15] D. Pisinger. Where are the hard knapsack problems? *Computers & Operations Research*, 32:2271–2284, 2005.

[16] N. J. Radcliffe. Genetic set recombination. In *Foundations of Genetic Algorithms*. Morgan Kaufmann Pulishers, 1993.

Table 3: Comparison of results for the RTR, R^3, RAR and TransRAR approaches in the portfolio selection problem.

Algorithm	Index	Best D	Success rate	Time (s)	Optimizations
GA-RTR	Hang Seng	**0.00321150**	0.93	802.7	$1.40 \cdot 10^7$
	DAX	2.77353490	0.36	1915.2	$2.83 \cdot 10^7$
	FTSE	2.00581544	0.39	3653.4	$5.52 \cdot 10^7$
	S&P	4.81011478	0.39	3938.1	$5.90 \cdot 10^7$
	Nikkei	1.00264537	0.25	3321.0	$5.11 \cdot 10^7$
GA-R^3	Hang Seng	**0.00321150**	0.93	802.7	$1.40 \cdot 10^7$
	DAX	2.83690197	0.40	1628.9	$2.81 \cdot 10^7$
	FTSE	1.97722629	0.40	3252.6	$5.50 \cdot 10^7$
	S&P	4.76271495	0.41	3597.2	$5.97 \cdot 10^7$
	Nikkei	1.03869098	0.25	3045.5	$5.20 \cdot 10^7$
GA-RAR $w = 1$	Hang Seng	**0.00321150**	1.00	539.1	$8.59 \cdot 10^6$
	DAX	**2.53162860**	1.00	2368.6	$3.12 \cdot 10^7$
	FTSE	**1.92150019**	0.95	4716.3	$6.09 \cdot 10^7$
	S&P	**4.69373181**	0.99	4931.9	$6.25 \cdot 10^7$
	Nikkei	**0.20197748**	1.00	7537.7	$7.18 \cdot 10^7$
GA-TransRAR	Hang Seng	**0.00321150**	1.00	497.6	$8.56 \cdot 10^6$
	DAX	**2.53162860**	1.00	1966.2	$3.11 \cdot 10^7$
	FTSE	**1.92150019**	1.00	3731.2	$6.07 \cdot 10^7$
	S&P	**4.69373181**	1.00	3912.8	$6.22 \cdot 10^7$
	Nikkei	**0.20197748**	1.00	4710.5	$7.15 \cdot 10^7$

[17] N. J. Radcliffe. The algebra of genetic algorithms. *Annals of Maths and Artificial Intelligence*, pages 339–384, 1994.

[18] N. J. Radcliffe and F. George. A study in set recombination. In *Proceedings of the 5th International Conference on Genetic Algorithms*. Morgan Kaufmann Pulishers, 1993.

[19] R. Ruiz-Torrubiano, S. Garcia-Moratilla, and A. Suarez. Optimization problems with cardinality constraints. In Y. Tenne and C. K. Goh, editors, *Computational Intelligence in Optimization: Implementations and Applications*. Springer Verlag, 2010.

[20] A. Schaerf. Local search techniques for constrained portfolio selection problems. *Computational Economics*, 20:177–190, 2002.

[21] J. Shapcott. Index tracking: genetic algorithms for investment portfolio selection. Technical report, EPCC-SS92-24, Edinburgh, Parallel Computing Centre, 1992.

[22] A. Simões and E. Costa. An evolutionary approach to the zero/one knapsack problem: Testing ideas from biology. In *Proceedings of the Fifth International Conference on Artificial Neural Networks and Genetic Algorithms ICANNGA*, 2001.