On the Importance of Phenotypic Duplicate Elimination in Decoder-Based Evolutionary Algorithms

Günther R. Raidl[1]
Institute of Computer Graphics
Vienna University of Technology
Karlsplatz 13/1861
1040 Vienna, Austria

Jens Gottlieb[2]
Department of Computer Science
Technical University of Clausthal
Julius-Albert-Str. 4
38678 Clausthal-Zellerfeld, Germany

Abstract

Premature convergence is a serious problem in many applications of evolutionary algorithms (EAs), since it decreases the EA's chance to reach new high-quality regions of the search space and hence degrades the overall performance. In particular decoder-based EAs are frequently susceptible to premature convergence due to their encoding redundancy. Our comparison of four decoder-based EAs for the multidimensional knapsack problem reveals the importance of maintaining the population's phenotypic diversity. We identify phenotypic duplicate elimination as a general method which efficiently prevents premature convergence for most EAs, while duplicate elimination on genotypic level is demonstrated as being unable to maintain phenotypic diversity.

1 Introduction

It is crucial for evolutionary algorithms (EAs) to maintain an appropriate balance between exploration and exploitation. If the ability of exploration is limited, an EA usually achieves a worse performance. In particular premature convergence prevents an EA from a successful search for optimal solutions. Decoder-based EAs are particularly susceptible to this problem, because the decoding procedure often induces a high redundancy between the genotypic search space and the phenotypic search space.

Four decoder-based EAs for the *multidimensional knapsack problem* (MKP), a well-known NP-complete combinatorial optimization problem with a wide range of applications (Garey and Johnson (1979)), are compared in our study. The MKP is stated as

$$\text{maximize} \quad \sum_{j \in J} p_j x_j \tag{1}$$

$$\text{subject to} \quad \sum_{j \in J} r_{ij} x_j \le c_i, \quad i \in I \tag{2}$$

$$x_j \in \{0,1\}, \qquad j \in J \tag{3}$$

with $I = \{1, \ldots, m\}$ and $J = \{1, \ldots, n\}$ denoting the sets of resources and items, respectively. Many exact and heuristic algorithms have been developed for the MKP and diverse variants (Chu and Beasley (1998), Martello and Toth (1990)), and in particular several EAs were proposed, see (Gottlieb (1999b)) for a survey.

The considered decoder-based EAs are tested with different duplicate elimination strategies. Duplicate elimination (DE) means that newly generated solutions are only accepted if they are not already contained in the current population. This is expected to be helpful for maintaining population diversity and, therefore, preventing premature convergence. Our experiments reveal that duplicate elimination at the genotypic level is clearly inferior to phenotypic duplicate elimination. The detailed investigation of the effects of duplicate elimination on the interplay of selection, crossover, mutation, and the employed decoder exhibits several important aspects which explain the impact of phenotypic duplicate elimination on the EA's performance. We expect similar results for decoder-based EAs applied to other problems.

A survey of the considered decoder-based EAs for the MKP is presented in Section 2, including a general comparison of different duplicate elimination strategies. Section 3 introduces several statistical measures which are useful to analyze the dynamics of the search. The effects of the different duplicate elimina-

[1]raidl@apm.tuwien.ac.at
[2]gottlieb@informatik.tu-clausthal.de

tion strategies on the considered EAs are discussed in detail in Section 4. Conclusions are given in Section 5.

2 Decoder-Based EAs for the MKP

The MKP has been used to compare several constraint handling techniques for EAs (Gottlieb (1999b)). Many approaches are based on penalizing infeasible solutions, however the best results are obtained when including heuristic repair and local optimization methods (Chu and Beasley (1998), Gottlieb (1999b), Raidl (1998)). While these EAs explicitly work in the *phenotyic search space* $P = \{0,1\}^n$, *decoder-based EAs* explore an arbitrary *genotypic search space* G, which is mapped into P by a *decoder* to perform an implicit search for MKP solutions. Such a decoder usually employs problem-specific knowledge to generate feasible solutions. The performance of decoder-based EAs strongly depends on the locality of the decoder and the employed operators and on the heuristic pressure, i.e. the ability to concentrate the search on promising phenotypes (Gottlieb and Raidl (1999)). Furthermore the decoder should be computationally fast since otherwise the fitness evaluation of the genotypes would be too time-consuming. Our experiments reveal that it is crucial to prevent premature convergence of the population. One way to achieve this would be to design special operators, however this obviously depends on the representation of G. Therefore we focus on the general concept of *duplicate elimination* (DE), which was found useful in several different EAs for the MKP (Chu and Beasley (1998), Gottlieb (1999a,b), Gottlieb and Raidl (1999), Hinterding (1994), Raidl (1998,1999)) and also for many other combinatorial optimization problems. Duplicate elimination means that newly generated solutions are rejected if they are already represented by the current population. There are two distinct interpretations of the duplicate concept: A newly generated genotype is called a duplicate, if (i) the genotype itself is already contained in the current population, or (ii) the corresponding phenotype is already represented by a genotype in the current population. These two interpretations lead to different DE strategies which we call *genotypic duplicate elimination* (case (i)) and *phenotypic duplicate elimination* (case (ii)). For comparison purposes, we also consider EAs without duplicate elimination and term the corresponding strategy *no duplicate elimination.* In the following, four decoder-based EAs are introduced and compared with respect to the effects of the different duplicate elimination strategies.

2.1 Permutation Based EA

The *permutation based EA* (PBEA) has been proposed by Hinterding (1994) for the (unidimensional) knapsack problem and can easily be adapted to the MKP (Raidl (1998), Thiel and Voss (1994)). A solution candidate is represented by a permutation $\pi : J \to J$ of the items. The decoder starts with the feasible solution $x = (0, \ldots, 0)$ and traverses all variables x_j in the order determined by π, increasing the corresponding variable from 0 to 1 if this does not violate any resource constraint. Hinterding employs standard permutation operators, namely uniform order based crossover and swap mutation which randomly exchanges two different positions.

2.2 Ordinal Representation based EA

The *ordinal representation based EA* (OREA) was originally considered for the traveling salesperson problem (TSP) by Grefenstette et. al (1985) however its application to MKP is straightforward (Gottlieb and Raidl (1999)). Solution candidates are represented by a vector v with $v_a \in \{1, \ldots, n-a+1\}$ for $a \in J = \{1, \ldots, n\}$. The decoder is based on a list initially containing all items in a predefined order and starts with the MKP solution $x = (0, \ldots, 0)$. Items are iteratively removed from the list and checked for inclusion in the solution. In detail, v is scanned from the first to its last position, interpreting each entry v_a as a position in the current list. Such a position identifies the next item j, for which x_j is increased to 1 if the resource constraints remain satisfied. Since each checked item is removed from the list, its size decreases by 1 during each step and reaches length 1 when the last item is to be selected. Classical one-point crossover is applicable because resulting offsprings always represent legal solutions. Moreover, a simple mutation operator is used which randomly chooses a position a and then draws v_a from $\{1, \ldots, n-a+1\}$. However, a closer look at the decoding procedure reveals that a change in a single position of v might have a major impact on the decoded solution since each item selection modifies the list, thus, influences all following item selections. OREA yields bad results for MKP, due to the weak locality of the decoder and the operators (Gottlieb and Raidl (1999)).

2.3 Surrogate Relaxation Based EA

Raidl (1999) proposed the *surrogate relaxation based EA* (SREA) which represents solution candidates by real-valued weights for the items. These weights are used to temporarily modify the profits p_j in the objective function (1) yielding a similar but slightly different MKP instance. This biased problem is then solved by a surrogate duality based heuristic. The solution obtained in this way is also feasible for the original, unbiased problem since the resource constraints (2) remain unchanged. The heuristic, which has originally been proposed by Pirkul (1987), starts with the solution $x = (0, \dots, 0)$ and traverses all items according to decreasing profit/pseudo-resource consumption ratio. Variables x_j are set to 1 if no resource constraint is violated. Pseudo-resource consumptions are determined via reasonable surrogate multipliers which are obtained from the linear programming (LP) relaxed MKP. Since this process would result in solving the LP relaxation for each solution candidate, Raidl (1999) suggests to determine the surrogate multipliers only once for the original problem in a preprocessing step to decrease the computational effort. SREA uses uniform crossover and a mutation operator which is applied 3 times to each new genotype, modifying a randomly chosen weight by resetting it to a new random value. The results obtained for SREA are the best among all decoder-based EAs for the MKP we are aware of.

2.4 Lagrangian Relaxation Based EA

The *Lagrangian relaxation based EA* (LREA) was also proposed by Raidl (1999) and is basically equivalent to SREA, except for the heuristic used to generate a solution for the biased problem. LREA employs the procedure introduced by Magazine and Oguz (1984) to obtain a solution via Lagrangian relaxation. Since the determination of exact Lagrange multipliers is too time-consuming, some reasonable (but usually suboptimal) multipliers are calculated by a simpler heuristic. Each obtained solution is then locally improved by traversing the variables according to decreasing profit and increasing them if feasibility can be maintained.

2.5 The Effects of Duplicate Elimination

We compare the described decoder-based EAs on one problem of the standard test suite of MKP benchmarks introduced in (Chu and Beasley (1998)) and available from the OR-Library[3]. Ten runs were performed for the considered instance of size $m = 10$, $n = 250$ and tightness ratio $\alpha = 0.5$ (which means that $c_i = \alpha \sum_{j \in J} r_{ij}$ holds for all $i \in I$). The test suite also contains problems of different size and α, however the selected instance is representative since the same basic trends were observed for the other problems too, coinciding with the results of our previous study (Gottlieb and Raidl (1999)). Our experiments are based on population size 100, parent selection via tournaments of size 2, steady-state replacement (replacing the worst individual), crossover probability 1.0 and an evaluation limit of 1 000 000 generated solutions. The results obtainted for the three duplicate elimination strategies (no DE, genotypic DE, phenotypic DE) and each EA are shown in Table 1. The solution quality is measured by the *gap* of the objective value concerning the optimal value of the LP-relaxed problem, i.e. $1 - max^{\mathrm{EA}}/opt^{\mathrm{LP}}$ with max^{EA} and opt^{LP} denoting the best objective value found by the EA and the optimal value of the LP relaxation of MKP, respectively. The *duplicate elimination ratio* R_{E} reflects the ratio of rejected duplicates among all generated solutions.

Table 1: Obtained gap [%] and total duplicate elimination ratio [%] for different EAs and DE strategies

DE	PBEA		OREA		SREA		LREA	
strategy	gap	R_{E}	gap	R_{E}	gap	R_{E}	gap	R_{E}
no	1.04	0.00	5.87	0.00	0.37	0.00	0.42	0.00
genotypic	1.02	0.00	5.47	1.45	0.37	0.00	0.46	0.00
phenotypic	0.33	2.88	1.80	31.94	0.24	3.52	0.27	1.39

Concerning the obtained quality, no DE and genotypic DE yield comparable gaps, while phenotypic DE results in a significantly better final solutions for each EA. The bad quality achieved by no DE and genotypic DE is caused by their inability to prevent premature convergence at the phenotypic level of the population. The similarity of no DE and genotypic DE is intuitive since no (PBEA, SREA, LREA) or few (OREA) newly generated solutions are rejected, hence both strategies yield a comparable behaviour. In general, R_{E} indicates the usefulness of phenotypic DE and the missing capability of genotypic DE to recognize phenotypic duplicates. A comparison of the different EAs reveals that genotypic DE recognizes duplicates only in the case of OREA. This shows that the operators employed by OREA tend to produce offsprings genotypically similar to their parents, which stays in accordance to our previous results (Gottlieb and Raidl (1999)). We proceed by a detailed examination of the interplay of operators, decoder, and duplicate elimination strategies and their effects on the EA dynamics.

3 Empirical Analysis of the Dynamics

To investigate the effects of the different duplicate elimination strategies concerning selection, crossover, and mutation separately, the following ratios were con-

[3] `http://mscmga.ms.ic.ac.uk/info.html`

tinuously observed during actual EA runs:

(a) *Duplicate elimination ratio* R_{E}: The ratio of detected and discarded duplicates among the total number of generated solutions.

Clearly, R_{E} is 0 if no duplicate elimination is applied. For genotypic and phenotypic duplicate elimination, it seems to be interesting at which stage of the evolutionary search how many solutions are discarded, because this might indicate the degree of premature convergence.

Note that the total duplicate elimination ratios were already presented in the last section.

(b) *Selection-duplicate ratio* R_{S}: The ratio of parent pairs selected for crossover, which are phenotypically identical.

In general, a high R_{S} indicates that the diversity of the population is small or parent selection is biased towards few solutions of extraordinary fitness. In case of the four considered EAs, tournament selection with tournament size 2 is used, yielding a moderate selection pressure. Therefore, a high R_{S} implies weak diversity.

(c) *Crossover-duplicate ratio* R_{C}: The ratio of offsprings generated by crossover that are phenotypically identical to one of their parents.

A high ratio R_{C} indicates that crossover is not able to produce offsprings that are sufficiently different from their parents, which leads to a reinforcement of premature convergence.

Usually, a high R_{S} implies a high R_{C}, since the traditional understanding of the behaviour of crossover is that as many phenotypic properties as possible should be inherited from parent solutions by an offspring (Goldberg (1989)). If R_{S} is low but R_{C} is nevertheless relatively high, crossover does not mix parental solutions in the context of phenotypic properties well enough.

(d) *Mutation-duplicate ratio* R_{M}: The ratio of mutations which do not cause changes in the represented phenotypes.

In general, R_{M} does not depend on the current population's diversity but influences it significantly. R_{M} can principally be controlled by tuning the mutation probability or rate. Obviously, a higher mutation rate would decrease this ratio, but this will also affect the EA to behave more like inefficient random search. Note that the aim of this paper is not to find a suitable mutation rate but to observe the properties of well-tuned mutation operators to either prevent or emphasize duplicate generation.

All these ratios were measured by performing 10 runs for each EA and each duplicate elimination strategy. Values were independently calculated for blocks of b consecutively generated solutions (generations), where $b = 10$ during the first 100 generations, $b = 100$ for generations 101 to 1 000, $b = 1\,000$ for generations 1 001 to 10 000, $b = 10\,000$ for generations 10 001 to 100 000, and $b = 100\,000$ for generations 100 001 to 1 000 000. Finally, results were averaged over the 10 runs per EA and duplicate elimination strategy. Figures 1 to 4 show plots for these average ratios, which are discussed in detail in the next section.

4 Discussion

For PBEA, SREA, and LREA, the ratio of eliminated duplicates R_{E} is always relatively small for all duplicate elimination (DE) strategies (never larger than 10%). But note that this observation does not imply that DE is less important for these EAs. The plots for R_{S} indicate that only phenotypic DE guarantees a reasonably small probability of selecting phenotypically identical solutions for crossover. While R_{S} is small in all cases up to generation 3 000, it thereafter increases dramatically up to nearly 100% when using genotypic or no DE, implying a substantial loss of diversity. Note that in case of phenotypic DE, R_{S} remains small but is nevertheless larger than 0, due to the possibility to select the same solution twice.

Clearly, R_{E} is 0 when using no DE. More interesting is the fact that R_{E} is also 0 when genotypic DE is applied in PBEA, SREA, and LREA. For OREA with genotypic DE, R_{E} is usually larger than 0 but nevertheless substantially smaller than in case of phenotypic DE. This means that the considered EAs generate genotypically identical solutions very seldom, and an EA with genotypic DE performs practically identically to the EA without DE.

This ineffectiveness of genotypic DE can be explained by the high encoding redundancy in all considered EAs due to the different sizes of G and P. While $|P| = 2^n$, the genotype search space has size $|G| = n!$ in PBEA and OREA, and is even larger for SREA and LREA since real values are allowed for each weight. Such a representational redundancy may decrease performance, but sometimes, as in case of SREA and LREA, it may also be beneficial and lead to better final results, see also (Ronald (1997)). An additional reason why different genotypes may often map to the same phenotypic solution is that the decoders may contain

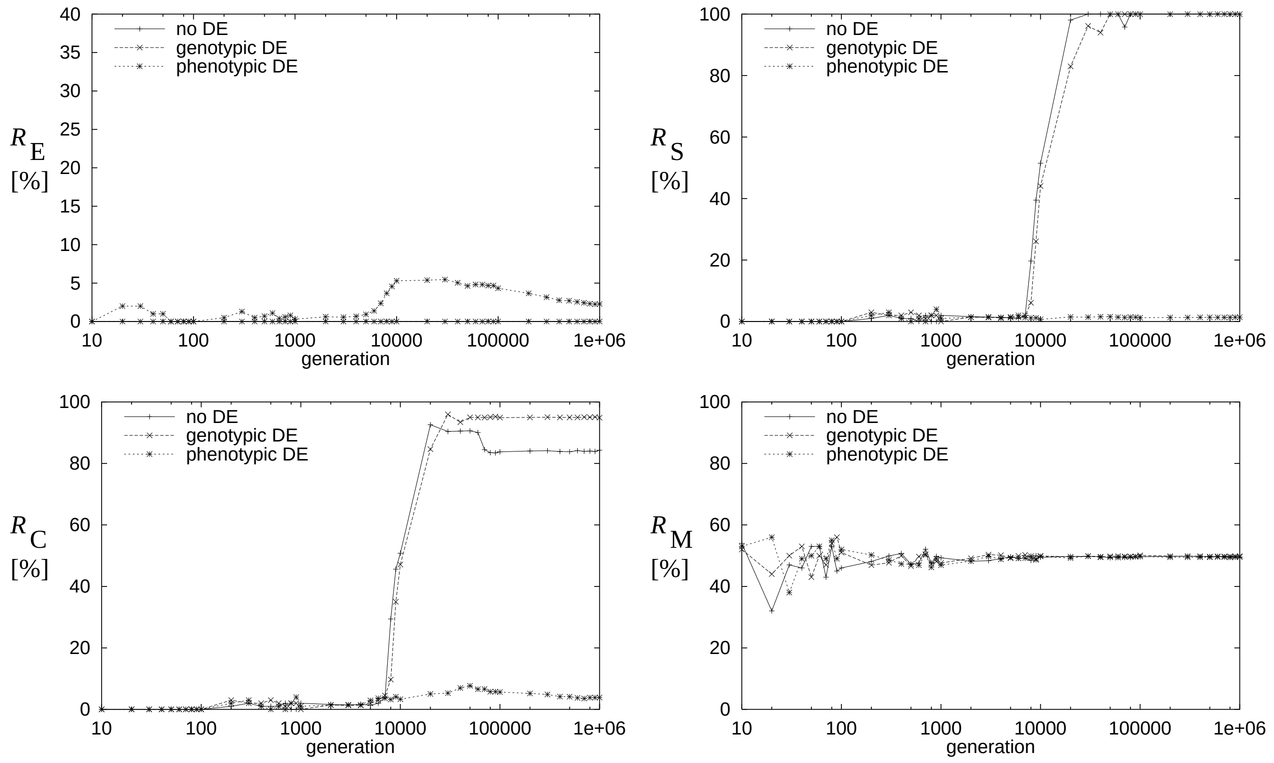

Figure 1: Results for PBEA

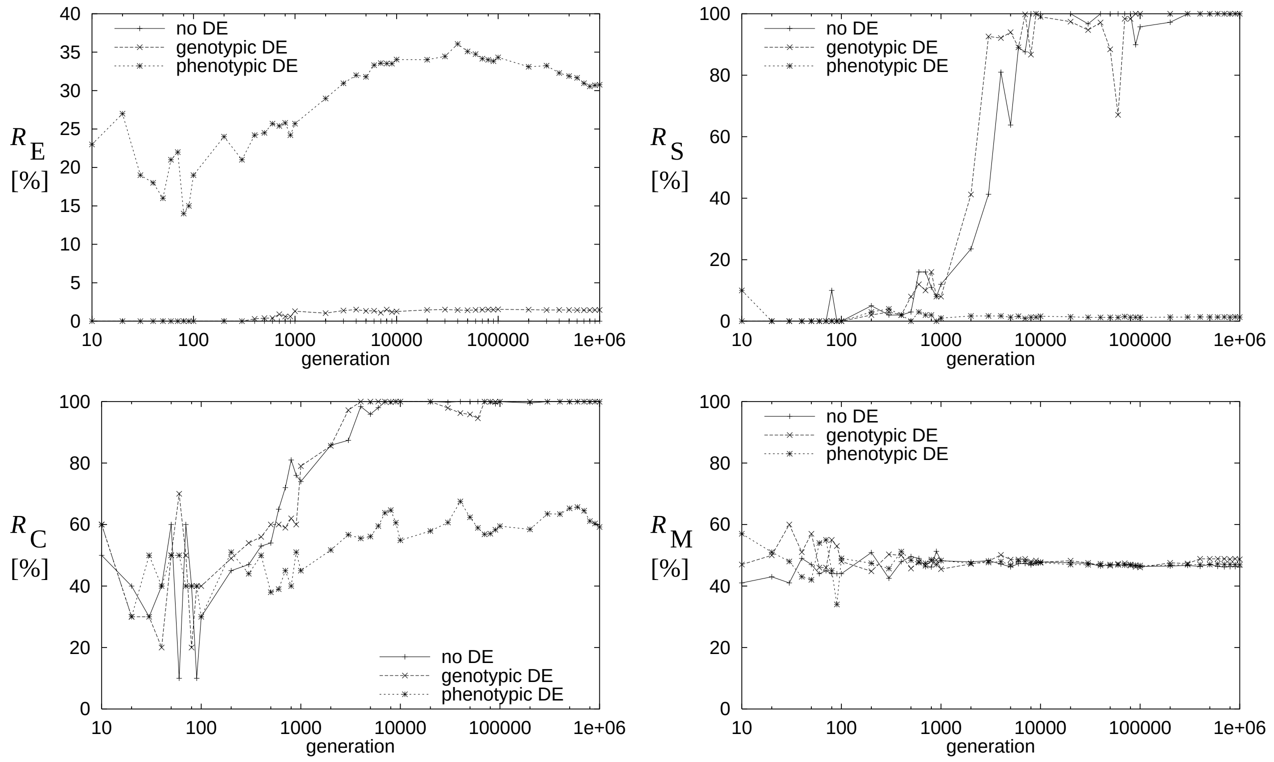

Figure 2: Results for OREA

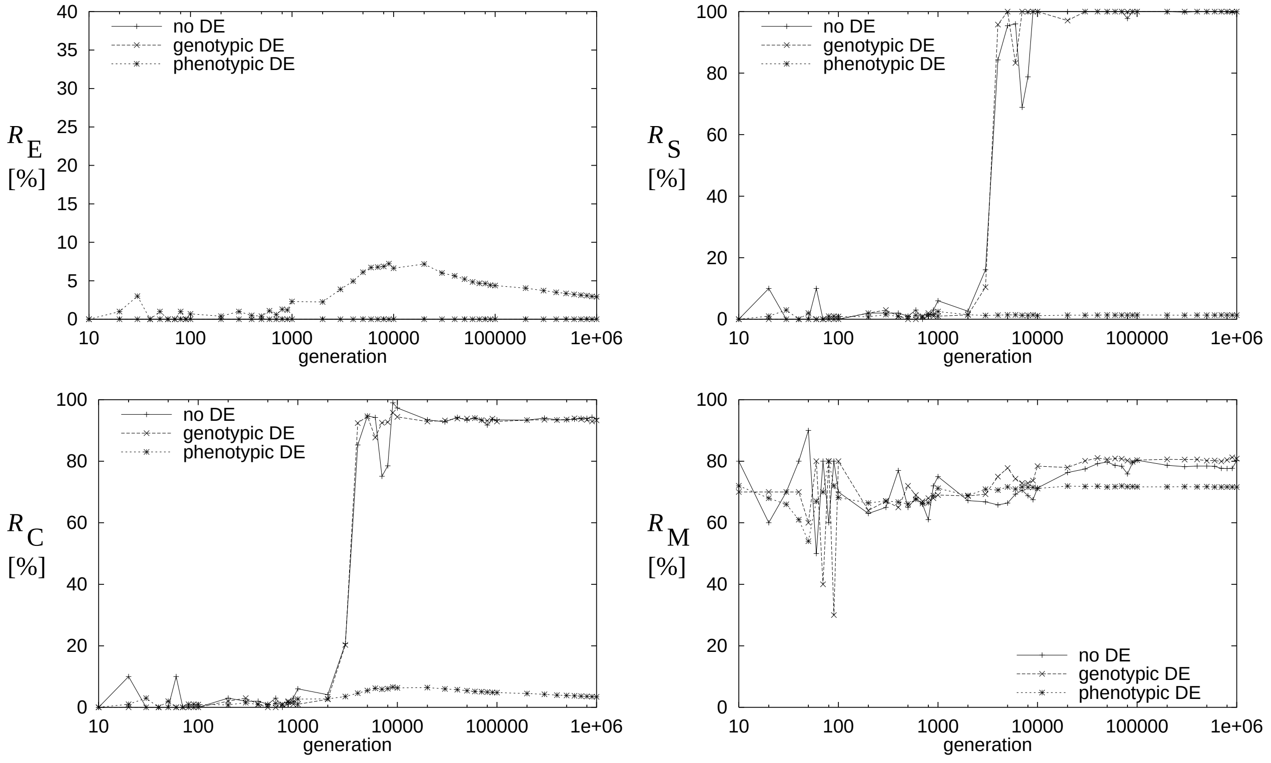

Figure 3: Results for SREA

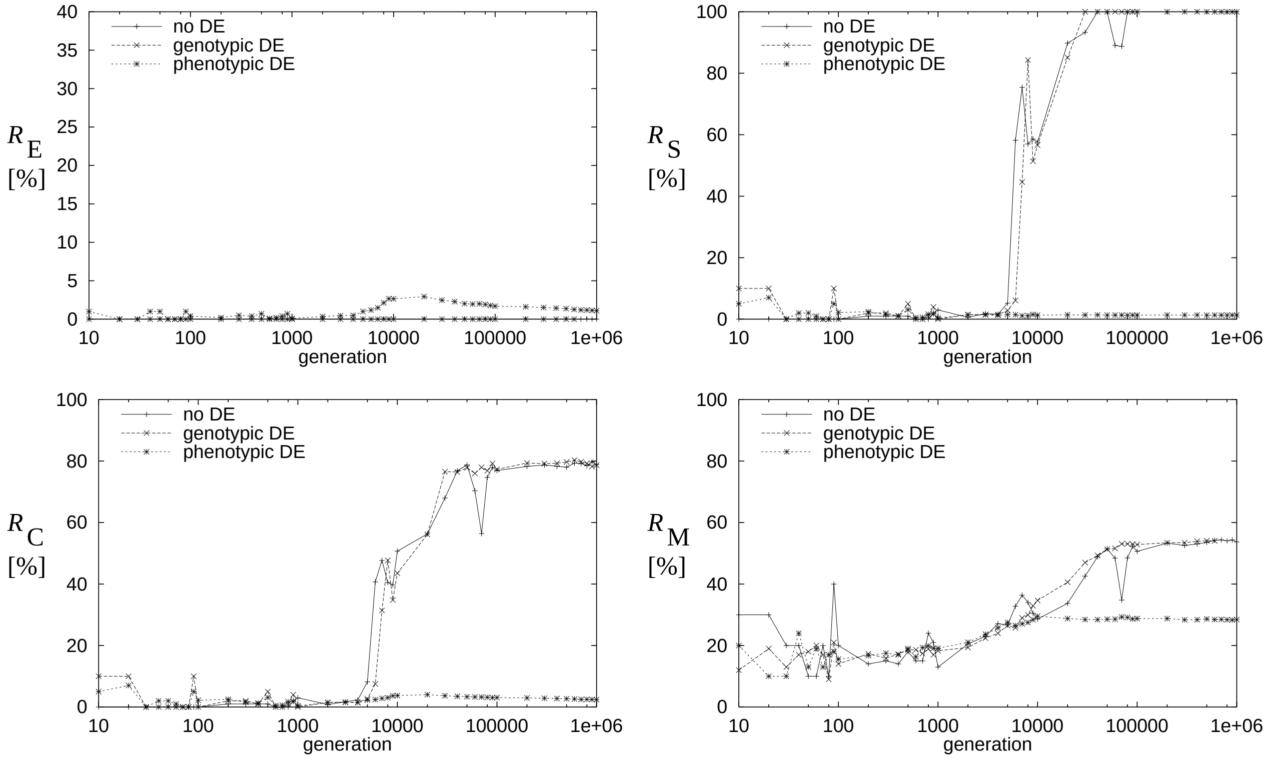

Figure 4: Results for LREA

local improvement techniques or heuristics which always or mostly lead to preferred phenotypes in a restricted subset $P' \subset P$. We call this effect *heuristic pressure* (Gottlieb and Raidl (1999)). Therefore, solutions $x \in P \backslash P'$ cannot be represented or have substantially smaller probabilities to be generated. While such a restriction of P to highly fit parts might sometimes be advantageous, it must be ensured that all promising areas and particularly the global optima are covered (Raidl (1999), Ronald (1997)). Note that all four EAs considered in this paper restrict the search to solutions on the boundary of the feasible region, which is known to contain the optimum (Gottlieb (1999a,b)). Furthermore, SREA and LREA have an even higher heuristic pressure since they additionally use heuristics depending on profits p_j and resource values r_{ij}.

When using phenotypic DE, R_{E} is low at the beginning of a run and increases significantly between generations 2 000 and 10 000. The obvious reason is the population's diversity which is initially high and decreases when an EA has found some locally nearly optimal solutions. Most noticeably is that R_{E} decreases again after about 20 000 generations. An explanation for this effect seems to be that an EA with the used replacement strategy finds relatively fast local optimal solutions lying relatively near to each other in search space. Thereafter, new better solutions, which are accepted by the replacement scheme for inclusion into the population, usually lie further away and therefore increase diversity again, which implies fewer generated duplicates in general.

Crossover-duplicate ratios R_{C} are strongly correlated to selection-duplicate ratios R_{S}. Therefore, crossover can only efficiently produce new solutions when the selected parents differ in their phenotypes. The loss of diversity above generation 3 000 in case of no DE or genotypic DE leads to an increase of R_{C} up to over 80% for all four EAs, which means that crossover does not perform efficiently any longer. For PBEA, SREA, and LREA with phenotypic DE, R_{C} remains below 10% during the whole run, while OREA already starts with a relatively high R_{C} ($\approx$ 40%). In case of phenotypic DE, this ratio increases only slightly, but when using no or genotypic DE, R_{C} soon reaches nearly 100%. The reason is that OREA uses one-point crossover, which might frequently exchange genes having no effect on the decoded phenotype, because the phenotypic properties are mainly determined by the first genes. Note that this is also the reason why the duplicate elimination ratio R_{E} is always relatively high for OREA with phenotypic DE.

The mutation-duplicate ratio R_{M} remains nearly constant at 50% in case of PBEA and OREA for all DE strategies. In case of PBEA this is due to the fact that only an exchange of one gene from approximately the first $\alpha \cdot n$ genes with another gene in the remainder results in a modification of the phenotype, which happens with a probability of about 50% for $\alpha = 0.5$. Similar considerations are valid for OREA.

In case of SREA and LREA, R_{M} does not remain constant but increases, especially when employing genotypic or no DE. The problem-dependent heuristics in these EAs seem to be responsible for this effect, since in general, these heuristics decrease the probability of good solutions to be modified by a single mutation in comparison to worse solutions.

5 Conclusions

We investigated the effects of using genotypic or phenotypic duplicate elimination in steady-state decoder-based EAs. Four EAs for the multidimensional knapsack problem were examplarily used for empirical studies. In general, phenotypic duplicate elimination turned out to be very important for good performance, since otherwise the crossover operator cannot reliably produce new solutions and the considered EAs get trapped at bad local optima very early. Results indicate that duplicate elimination based on comparing genotypes cannot avoid or reduce this premature convergence. Only a duplicate elimination which avoids genotypes representing the same phenotypic solutions in the population can reliably achieve this. A reason for the failure of genotypic duplicate elimination is that the considered EAs use large genotype search spaces which are mapped to significantly smaller parts of the actual phenotype space. This implies a high encoding redundancy, which is typical for decoder-based EAs applied to combinatorial optimization problems. Therefore our results should also be valid for other decoder-based EAs. In this light, other approaches to maintain higher diversity, e.g. niching techniques like deterministic crowding, might also be beneficial and influence performance significantly. Closer investigations should therefore be performed.

References

P. C. Chu and J. E. Beasley (1998): *A Genetic Algorithm for the Multidimensional Knapsack Problem*, Journal of Heuristics 4, pp. 63 – 86

M. D. Garey and D. S. Johnson (1979): *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, San Francisco

D. E. Goldberg (1989): *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley

J. Gottlieb (1999a): *Evolutionary Algorithms for Multidimensional Knapsack Problems: The Relevance of the Boundary of the Feasible Region*, to appear in Proc. of the Genetic and Evolutionary Computation Conf., Orlando, FL

J. Gottlieb (1999b): *On the Effectivity of Evolutionary Algorithms for Multidimensional Knapsack Problems*, submitted to Evolution Artificielle, Dunkerque, France

J. Gottlieb and G. R. Raidl (1999): *Characterizing Locality in Decoder-Based EAs for the Multidimensional Knapsack Problem*, submitted to Evolution Artificielle, Dunkerque, France

J. J. Grefenstette, R. Gopal, B. Rosmaita, and D. Van Gucht (1985): *Genetic Algorithms for the Traveling Salesman Problem*, in Proc. of the 1st Int. Conf. on Genetic Algorithms, Hillsdale, NJ, pp. 160 – 168

R. Hinterding (1994): *Mapping, Order-independent Genes and the Knapsack Problem*, in Proc. of the 1st IEEE Int. Conference on Evolutionary Computation, Orlando, FL, pp. 13 – 17

M. J. Magazine and O. Oguz (1984): *A Heuristic Algorithm for the Multidimensional Zero–One Knapsack Problem*, European Journal of Op. Res., 16, pp. 319 – 326

S. Martello and P. Toth (1990): *Knapsack Problems: Algorithms and Computer Implementations*, J. Wiley & Sons

H. Pirkul (1987): *A Heuristic Solution Procedure for the Multiconstrained Zero-One Knapsack Problem*, Naval Research Logistics 34, pp. 161 – 172

G. R. Raidl (1998): *An Improved Genetic Algorithm for the Multiconstrained 0–1 Knapsack Problem*, in Proc. of the IEEE Int. Conf. on Evolutionary Computation, Anchorage, AL, pp. 207 – 211

G. R. Raidl (1999): *Weight-Codings in a Genetic Algorithm for the Multiconstraint Knapsack Problem*, to appear in Proc. of the Congress on Evolutionary Computation, Washington DC

S. Ronald (1997): *Robust Encodings in Genetic Algorithms*, in D. Dasgupta, Z. Michalewicz (eds.), Evolutionary Algorithms in Engineering Applications, pp. 29 – 44, Springer

J. Thiel and S. Voss (1994): *Some Experiences on Solving Multiconstraint Zero-One Knapsack Problems with Genetic Algorithms*, INFOR 32, pp. 226 – 242