Heterogeneous Constraint Handling for Particle Swarm Optimization

Ludmila Omeltschuk, Sabine Helwig, Moritz Mühlenthaler, Rolf Wanka
Department of Computer Science
University of Erlangen-Nuremberg, Germany
Email: ludmila_omeltschuk@web.de, {sabine.helwig, moritz.muehlenthaler, rwanka}@cs.fau.de

***Abstract*—We propose a generic, hybrid constraint handling scheme for particle swarm optimization called Heterogeneous Constraint Handling. Inspired by the notion of social roles, we assign different constraint handling methods to the particles, one for each social role. In this paper, we investigate two social roles for particles, 'self' and 'neighbor.' Due to the usual particle dynamics, a powerful mixture of the two corresponding constraint handling methods emerges. We evaluate this heterogeneous constraint handling approach with respect to the complete set of the CEC 2006 benchmark instances. Our results indicate that a such a heterogeneous combination of two constraint handling methods often leads to significantly better results than running each individual constraint handling method separately and returning the best solution obtained.**

I. Introduction

Particle swarm optimization (PSO) is a popular technique for solving continuous optimization problems, which is inspired by cooperative collective animal behavior [11]. It has been applied successfully to a wide range of continuous optimization problems, and has more recently been adapted to discrete problems [1], [3], [9], [12], [24]. In PSO, every individual from the population determines its new position depending on its own historically best position and on the best positions of its fellow neighbors (the mathematical details are presented in Section II-B). Generally in PSO, the exploration of potential solutions is restricted to an n-dimensional box. The effects of different approaches for handling box constraints (i. e., what to do if a particle tries to leave the allowed space) on the PSO behavior have been analyzed comprehensively in [8]. However, since optimization problems related to real-world applications typically have more complex and possibly non-linear constraints, there has been some effort to include suitable constraint handling methods in PSO.

Currently, there exist two kinds of approaches for incorporating sophisticated constraint handling in PSO. Approaches of he first kind build on well-established methods from evolutionary algorithms and related research, such as *exterior penalty* methods [4]. In contrast to these approaches, the constraint handling method proposed by Toscano Pulido and Coello Coello [25] is specifically designed for PSO and integrates directly in the PSO dynamics. Our proposed heterogeneous constraint handling scheme fills the gap between these two kinds of approaches by integrating combinations of any two constraint handling methods in the PSO dynamics in a natural manner. Similarly to the PSO algorithm itself, our idea for the proposed constraint handling method is inspired by the social behavior of individuals. Similarly to an individual which assesses itself differently than its peers, a particle may apply a different method for judging its own best position than for judging the best positions of its neighbours.

In our heterogeneous approach, each particle uses two different constraint handling methods in order to update its postion. A hybrid combination of the two methods then emerges through the communication between particles in the swarm. We evaluate several combinations of exterior penalty methods and also include the approach by Toscano Pulido and Coello Coello in our hybridization scheme. Our results show that a heterogeneous combination of two constraint handling methods often significantly outperforms each individual constraint handling method. This phenomenon is also well-known in the area of discrete combinatorial optimization. For instance, the hybridization of different algorithms for Max-SAT may lead to a better overall algorithm because the worst case of the combined algorithm is better than the worst cases of the single algorithms (e. g., see [23, pp. 456ff]).

The remainder of this paper is organized as follows. In the next section we provide the problem formulation as well as relevant background information on PSO and constraint handling methods. Section III describes in detail how heterogeneous constraint handling works and how it integrates in the standard PSO algorithm. Section IV presents an evaluation of Heterogeneous Constraint Handling based on the complete CEC 2006 benchmark instances. We provide a qualitative comparison of the performance of a number of heterogeneous and homogeneous constraint handling methods using the Wilcoxon rank-sum test as well as quantitative data based on the mean and standard deviation of the objective values. Section V gives a summary and a brief outlook to conclude this work.

II. Background

A. Problem Formulation

Without loss of generality, we are dealing with minimization problems of the form

$$\begin{aligned} \min \quad & f(x) \\ \text{s. t.} \quad & g_j(x) \leq 0, \quad j \in \{1,\ldots,l\} \\ & h_k(x) = 0, \quad k \in \{1,\ldots,m\}, \end{aligned} \tag{1}$$

inside an n-dimensional rectangle $B \subset \mathbb{R}^n$, where $x \in B$, f is the real-valued objective function, g_j are the inequality constraints and h_k are the equality constraints. No further assumptions are made about the nature of the constraint functions and the objective function, i. e., any choice of linear and non-linear

978-1-61284-052-9/11/$26.00 ©2011 IEEE

functions is suitable. The feasible part of the search space $F \subseteq B$ consists of all points satisfying all constraints.

B. Particle Swarm Optimization

Particle swarm optimization is a population-based metaheuristic inspired by the simulation of social behavior [11]. A swarm of candidate solutions called particles cooperatively solves instances of non-linear optimization problems such as (1) by letting particles iteratively explore the search space and exchange information about the best solutions they have encountered so far.

Each particle i in the swarm has a position vector $x_{i,t}$ and a velocity vector $v_{i,t}$, where t is the iteration number. Additionally, each particle remembers the position $p_{i,t}$ corresponding to the best objective value it has encountered so far, the so-called *private guide*. Furthermore, each particle is connected to a set of other particles called neighbors. The best solution found in a particle's neighborhood is referred to as the *local guide* $l_{i,t}$. By selecting a local guide for each particle, information about the search space is exchanged between particles in a common neighborhood. Different neighborhood topologies have been studied in the literature, such as the *Lbest*, *Gbest*, and *von Neumann* topologies (see [11], [13], [17]). In each iteration t, the positions of all particles are updated according to the PSO movement equations. The new position of particle i is computed as follows:

$$\begin{aligned} v_{i,t} &= \omega \cdot v_{i,t-1} + U[0,c_1] \odot (p_{i,t-1} - x_{i,t-1}) \\ &\quad + U[0,c_2] \odot (l_{i,t-1} - x_{i,t-1}) \qquad (2) \\ x_{i,t} &= x_{i,t-1} + v_{i,t} \qquad (3) \end{aligned}$$

$\odot$ denotes element-by-element vector multiplication, ω (the *inertia weight*), c_1, and c_2 (the *acceleration coefficients*) denote constants chosen by the user, and $U[a,b]$ is an n-dimensional random vector with components drawn uniformly at random from $[a,b]$.

After updating the positions, each particle i updates its private guide if the new position $x_{i,t}$ yields a better objective value than $p_{i,t-1}$. Similarly, all local guides are updated.

In the following, we drop the iteration number t from the indices.

C. Constraint Handling Methods

Michalewicz and Schoenauer provided a classification of common constraint handling approaches into the following four categories [19]:

1) Feasibility-preserving methods
2) Penalty-based methods
3) Methods based on the separation of feasible and infeasible solutions
4) Hybrid methods

Our proposed heterogeneous constraint handling approach combines various methods from categories 2 and 3. Hence, it falls in the category of hybrid methods. Constraint handling methods from the categories 2 and 3 typically provide some mechanism for comparing the quality of feasible and infeasible solutions. The penalty-based methods replace the original objective function with a new one, which applies a penalty to infeasible solutions. This means, penalty-based methods provide a way to quantify how much worse infeasible solutions are compared to feasible ones. In contrast, methods from category 3 typically provide a comparison operator, which takes the (in)feasibility of the solutions to be compared into account. The outcome of the comparison is a purely qualitative judgment of the relative quality of two solutions.

Feasibility-preserving constraint handling methods (category 1) naturally require problem-dependent knowledge which is harnessed by specialized solution representations and suitable mutation/recombination operators [14], [15], [18], [20]. Although feasibility-preserving methods are often very effective, they are not applicable to general problem formulations such as (1) and are thus not included in our evaluation of heterogeneous constraint handling. For particular problems however, it might be beneficial to hybridize such methods with those from the other categories using Heterogeneous Constraint Handling.

Constraint handling methods from category 2 assign a penalty to the objective values of infeasible candidate solutions. This means, for updating the private and the local guide of a particle, the objective function $f(x)$ is replaced with the penalty function $\phi(x)$. Different flavors of penalty functions have been explored in the literature, such as *static* [22], *dynamic* [10], [26], *adaptive* [5], [7], *death penalty* [4] as well as the penalty function proposed by Deb in [6]. For an in-depth discussion of these approaches, please refer to [4]. For our evaluation of the heterogeneous constraint handling, *static* and *dynamic penalty* functions as well as Deb's penalty function from [6] were chosen from category 2 since they are well-studied and feature a comparatively low number of parameters.

In the *static penalty* scheme, the amount penalty is proportional to the number of violated constraints [22]:

$$\phi_{static}(x) = \begin{cases} f(x) & \text{if } x \in F, \\ K \cdot \left(1 - \dfrac{s}{l+m}\right) & \text{otherwise,} \end{cases}$$

where s is the number of satisfied constraints and K is some sufficiently large constant such that there is a clear distinction between feasible and infeasible solutions. Please note that the objective function only needs to be evaluated if the solution if feasible.

The *dynamic* penalty scheme by Joines and Houck [10] differs from the *static* penalty in two ways: First, the amount of penalty depends on the extent of the constraint violations. And second, an increasing amount of penalty is applied as the search progresses, in order to avoid a penalty value which is too low if the constraints are violated only slightly:

$$\phi_{dynamic}(x) = f(x) + (C \cdot t)^{\alpha} \left(\sum_j \max\{0, g_j(x)\}^{\beta} + \sum_k D_k(x) \right),$$

with

$$D_k = \begin{cases} 0 & \text{if } |h_k(x)| \leq \varepsilon, \\ |h_k(x)| & \text{otherwise,} \end{cases}$$

where t is the iteration number, and α, β and C are constants which can be tuned to the instance at hand.

The parameterless constraint handling method proposed by Deb in [6] falls into categories 2 and 3. It has originally been proposed for a genetic algorithm in conjunction with a selection mechanism favoring feasible solutions, but it can be employed in other population-based approaches such as PSO. The objective values of the candidate solutions are adjusted according to $\phi_{Deb}(x)$:

$$\phi_{Deb}(x) = \begin{cases} f(x) & \text{if } x \in F, \\ f_{worst} + \sum_j g_j(x) + \sum_k |h_k(x)| & \text{otherwise.} \end{cases}$$

A penalty is applied to infeasible solutions depending on the extent of the constraint violations. Making the penalty relative to the worst individual of the population (f_{worst}) guarantees the any feasible solution of the current population is better than any infeasible one. Considering just the objective values, this means feasible and infeasible solutions are kept in separate groups, which is characteristic to constraint handling methods in category 3.

In contrast to the penalty-based methods, the constraint handling approach proposed by Toscano Pulido and Coello Coello in [25] influences the direction of the search of a particle swarm by providing a specialized comparison operator for local/global guide selection favoring feasible particles. Hence, this approach falls in the third category of the above classification. Whenever the quality of two particles is compared, the following rules apply:

1) If both particles are feasible, select one with the lowest objective value
2) If one particle is feasible and the other is infeasible, select the feasible one
3) If both particles are infeasible, select one with the lowest total amount of constraint violations (normalized w.r.t. the largest constraint violation in the population)

This approach will be referred to as constraint handling strategy *super*. In contrast to the implementation by Toscano Pulido and Coello Coello, no turbulence operator was added to the PSO algorithm, since our aim is to make a fair comparison of different constraint handling method on a standard PSO.

III. Heterogeneous Constraint Handling

In this section, we describe our new approach. Heterogeneous Constraint Handling introduces social roles for particles, and for each social role there is an individual constraint handling method. In our proposed approach we use two different social roles for each particle, which we refer to as *self* and *neighbor*. The constraint handling mechanism for the *self* role influences the selection of a particle's private guide. Similarly, the constraint handling mechanism for the *neighbor* role determines, which solutions are provided as local guides to a particle's neighbors. Hence for a particle swarm, a hybrid constraint handling mechanism emerges, which combines the constraint handling methods of the *self* and *neighbor* roles.

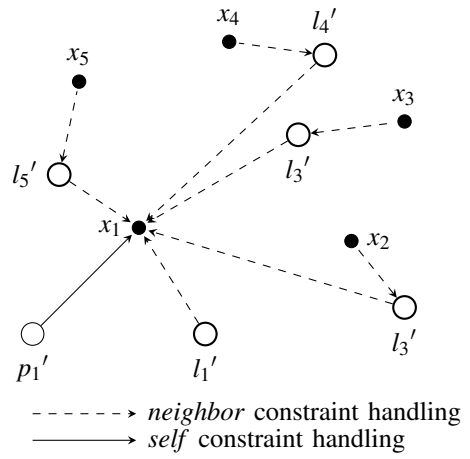

Fig. 1. Heterogeneous Constraint Handling: propagation of constraint handling information from the point of view of P_1

In contrast to a traditional PSO (see section II-B), each particle i keeps track of *two* personal best solutions p_i' and l_i'. p_i' is the best solution encountered by particle i so far according to the *self* constraint handling method. Similarly, l_i' is the personal best solution of particle i according to the *neighbor* constraint handling strategy. p_i' serves as private guide for the particle i in equation 2 and l_i' is provided to its neighbors as a potential local guide. The local guide of particle i is chosen at random from the *best* solutions in its neighborhood with respect to the *neighbor* constraint handling method.

Figure 1 illustrates by example how constraint handling information is propagated between five particles in a neighbourhood from the point of view of particle 1 during its position update. Please note that particle 1 is included in its own neighborhood. The new position of particle 1 is determined as usual by the equations (2) and (3). When updating the velocity v_1, the particle's private guide is p_1', which has been chosen according to the *self* constraint handling strategy. Its local guide is is chosen at random from the *best* solutions in $\{l_d'\}_{1 \leq d \leq 5}$. After the position update, the two personal best solutions of particle 1 are updated according to the *self* and *neighbor* constraint handling strategies, respectively.

Ideally, for such a hybrid combination of different constraint handling strategies, one constraint handling method compensates for the weaknesses of the other and vice versa, such that there is an improvement compared to each individual constraint handling method. Indeed, the results presented in the next section indicate that heterogeneous constraint handling is more likely to produce feasible solutions and also improves on the objective values in comparison to the homogeneous approaches reviewed in section II-C. As possible direction for future research, more than two constraint handling could be hybridized via Heterogeneous Constraint Handling by assigning different private guide constraint handling methods to

different particles.

IV. Results

Our proposed Heterogeneous Constraint Handling is a generic mechanism for creating a new constraint handling method from virtually any choice of two individual constraint handling mechanisms suitable for the problem at hand. For evaluating the performance of the Heterogeneous Constraint Handling, various combinations of the *static*, *dynamic*, *Deb* and *super* constraint handling methods, described in section II-C, were created. This evaluation however can only consider a fraction of the many possible combinations of different constraint handling strategies which can be created using our scheme. There are many more promising constraint handling strategies in the literature waiting to be hybridized. Our purpose here is to show that creating heterogeneous combinations is beneficial in many cases for the homogeneous strategies under consideration.

The heterogeneous methods as well as the individual homogeneous strategies were tested on all 24 instances ($g01,\ldots,g24$) of the CEC 2006 benchmark [16]. For each instance and each constraint handling method, 50 independent runs were performed. We provide both, a qualitative comparison of the performance of the constraind handling methods using the Wilcoxon rank-sum test (see tables II and III), and quantitative data based on the mean and standard deviation of the objective values (see tables IV and V). Our results indicate that a heterogeneous combination of two constraint handling methods often outperforms both individual constraint handling methods. It also turns out that heterogeneous constraint handling is *asymmetric* in the sense that it is quite sensitive to swapping private and local constraint handling strategies. For instance, using *static* penalty as private strategy and *dynamic* penalty as local strategy outperforms both *static* and *dynamic* as individual strategies. However, swapping the private and local strategies shows no improvement compared to the homogeneous approaches.

Our results were obtained using the PSO configuration shown in Table I. Please note that in our experiments each particle is included in its neighborhood as indicated in figure 1. The maximum number of evaluations performed refers to the number of problem evaluations with respect to a particle, which potentially includes the evaluation of the objective function, the constraints, or both. For equality constraints, a tolerance level $\varepsilon = 10^{-6}$ was used in order to avoid floating point precision/comparison issues. The penalty factor K for ϕ_{static} was set to 10^9 following [21]. The parameters C, α and β for the *dynamic* penalty function were determined by experimentation with the values suggested in [10].

Table II shows a qualitative comparison of the constraint handling methods *static*, *dynamic*, *static*×*dynamic*, *static*×*Deb*, and *dynamic*×*static*. Each row and column corresponds to a homogeneous or heterogeneous constraint handling method. For the heterogeneous methods, the name of the *self* strategy appears first followed by the name of the *neighbor* strategy. The heterogeneous strategy *static*×*dynamic* for example uses static as *self* and dynamic as *neighbor* strategy. Each cell shows the instances for which the row constraint handling method performs significantly better than the column strategy. The Wilcoxon rank-sum test (see [27]) was employed with a significance level of 1% to determine the contents of each cell. For infeasible runs, an objective value of 10^{10} was assumed.

Parameter	Value
ω	0.72984 [2]
c_1, c_2	1.496172 [2]
Number of particles	49
Number of evaluations	340,000 [25]
Neighborhood topology	*von Neumann*
Neighborhood size	5
ε	10^{-6}
K	10^9 [21]
C	4.5
α	1.0
β	2.0

TABLE I
PSO and constraint handling configuration for the experiments

The results in table II show that the heterogeneous strategy *static*×*dynamic* outperforms the homogeneous *static* strategy for 5 instances and the *dynamic* strategy for 6 instances. On the other hand *static* is better than *static*×*dynamic* for g04 and g06, and *dynamic* never outperforms the heterogeneous method. Please note that while neither the *static* nor the *dynamic* penalty method produce any feasible solutions for instances g05, g13 and g17, the heterogeneous version has a rate of 52%, 100% and 10% feasible runs, respectively (see Table IV). For an intuitive explanation, why a combination of static and dynamic penalty schemes works so well, please consider the amount of penalty dealt by the *static* and *dynamic* constraint handling methods for a highly constrained problem as the search progresses. As shown in Figure 2 the search can be divided in two phases. In the first phase, the *static* penalty function dominates due to the large number of violated constraints and the large factor K. In the later stages of the search however, many particles are likely to have small (but equal) number of violated constraints and thus are assigned the same objective value. Thus, the static penalty scheme is not sufficient for identifying promising regions of the search space. However, the *dynamic* penalty scheme considers the extent of constraint violations and can thus complement the static penalty and drive the particles towards feasible regions of the search space. This observation however does not consider the flow of information within the swarm and hence cannot explain asymmetric performance of Heterogeneous Constraint Handling when swapping *self* and *local* constraint handling strategies (see *static*×*dynamic* versus *dynamic*×*static* in Table II).

Another example of a good combination of constraint handling methods is *static*×*Deb*. Table II indicates that despite the relatively poor performance of the homogeneous *Deb* strategy, the combination *static*×*Deb* performs very well. In fact, it

	static	*dynamic*	*Deb*	*static*×*dynamic*	*static*×*Deb*	*dynamic*×*static*
static		g04, g06, g10	g07, g09, g10, g11, g19	g04, g06	g07, g09, g11, g19	g04, g06, g10
dynamic	g11, g15		g07, g09, g11, g15, g19		g07, g09, g11, g19	g11, g15
Deb	g03, g14, g23	g03, g04, g06, g14, g23		g03, g04, g06, g14, g23		g03, g04, g06, g14, g23
static×*dyn.*	g05, g10, g11, g13, g15	g04, g05, g6, g10, g13, g15	g05, g07, g9, g10, g11, g13, g15, g19		g07, g09, g10, g11, g19	g04, g05, g06, g10, g11, g13, g15
static×*Deb*	g03, g05, g10, g13, g14, g15, g17, g23	g03, g04, g05, g06, g10, g13, g14, g15, g17, g18, g23	g03, g05, g07, g09, g10, g11, g13, g15, g17, g19, g23	g03, g04, g05, g06, g13, g14, g17, g23		g03, g04, g05, g06, g10, g13, g14, g15, g17, g23
dyn.×*static*			g07, g09, g11, g19		g07, g09, g11, g19	

TABLE II
COMPARISONS OF DIFFERENT HETEROGENEOUS CONSTRAINT HANDLING METHODS AND THEIR HOMOGENEOUS COUNTERPARTS.

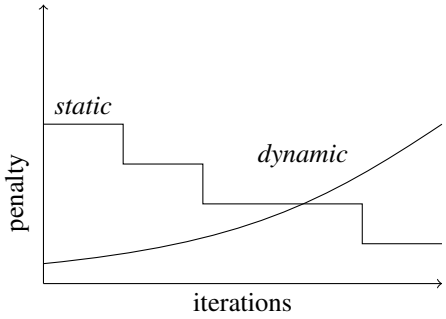

Fig. 2. An illustration of the penalty dealt by a *static* and a *dynamic* penalty function during a run of the PSO algorithm.

outperforms both individual strategies since *static*×*Deb* is significantly better than the homogeneous *static* strategy for 8 instances and better than the homogeneous *Deb* strategy for 11 instances. Also, *static*×*Deb* is better than *static*×*dynamic* strategy on 8 of the 24 instances, the latter being the better choice on 5 instances.

The mean objective values in Tables IV and V show that the *super* strategy by Toscano Pulido and Coello Coello [25] has the best overall performance of all homogeneous strategies under consideration. This can be explained by the fact that all instances except g08, g12, and g19 have at least one active constraint, i. e., for most instances the global optimum lies just on the boundary between F and $B \setminus F$. The *super* strategy avoids the choice of the "right" amount of penalty altogether since infeasible solutions are implicitly penalized when they are compared to feasible ones. Additionally, the constraint violation is normalized with respect to the largest constraint violation in the population. This implies that constraints which are hard to satisfy do not have as much impact on the evaluation as the easy constraints. Hence, the boundary between feasible and infeasible regions of the search space can be explored better than with the penalty-based approaches. Table III shows the significance analysis for the homogeneous *super* strategy and several heterogeneous combinations involving the *super* strategy. Clearly, none of the heterogeneous methods can significantly improve on the *super* strategy with the exception of *super*×*dynamic*, which is better on g10. This puts the the quantitative data in table V into perspective: Although table V indicates that there are improvements of the mean objective values compared to the homogeneous *super* strategy for a variety of instances, the Wilcoxon rank-sum test shows that they are not statistically significant, at least for the significance level of 1%.

V. CONCLUSIONS

We proposed Heterogeneous Constraint Handling, a general scheme for hybridizing different constraint handling methods naturally in PSO. In our approach we consider two social roles for each particle, *self* and *neighbor*, and use a separate constraint handling mechanism for each role. The *self* and *neighbor* constraint handling strategy determine the selection of a particle's private and local guide, respectively. A hybrid constraint handling method emerges via the usual PSO dynamics.

Our evaluation of Heterogeneous Constraint Handling is based on the complete set of CEC 2006 benchmark instances [16]. Our results indicate, that heterogeneous combinations of two different constraint handling methods often outperform each individual constraint handling method. It turns out however, that the performance of Heterogeneous Constraint Handling is quite sensitive to swapping constraint handling mechanisms for the private and local guides. Hence, care needs to be taken when selecting the individual constraint handling strategies.

Due to the fact that heterogeneous strategies often outperform the individual constraint handling mechanisms, creating a hybrid combination of more than two constraint handling mechanisms to further improve the performance seems to be a promising topic for future research.

REFERENCES

[1] D. Anghinolfi and M. Paolucci. A new discrete particle swarm optimization approach for the single-machine total weighted tardiness scheduling problem with sequence-dependent setup times. *European Journal of Operational Research*, 193(1):73–85, February 2009.

[2] D. Bratton and J. Kennedy. Defining a Standard for Particle Swarm Optimization. In *Proc. 2007 IEEE Swarm Intelligence Symposium (SIS)*, pages 120–127, 2007.

[3] M. Clerk. Discrete particle swarm optimization, illustrated by the traveling salesman problem. In *New Optimization Techniques in Engineering*, pages 219–239. Springer, 2004.

	super	*static×super*	*super×static*	*super×dynamic*	*super×Deb*
super		g05, g22, g23	g05, g13, g14, g15, g17, g21, g22, g23	g03, g04, g05, g06, g13, g14, g17, g21, g22, g23	g07, g09, g10, g11, g19, g21, g22
static×super			g03, g05, g13, g14, g15, g17, g21, g23	g03, g04, g06, g13, g14, g17, g21, g23	g07, g09, g10, g11, g18, g19, g21
super×static				g03, g04, g06, g14, g18	g07, g09, g10, g11, g18, g19
super×dyn.	g10	g10	g05, g10, g13, g15, g17, g23		g07, g09, g10, g11, g19
super×Deb		g05, g14	g05, g13, g14, g15, g17, g21, g23	g03, g04, g05, g06, g13, g14, g17, g21, g23	

TABLE III

COMPARISONS OF THE HOMOGENEOUS *super* STRATEGY WITH VARIOUS HETEROGENEOUS METHODS INVOLVING THE *super* STRATEGY FROM [25].

	Static	Dynamic	Deb	Static×Dynamic	Static×Deb
$g01$	$-14.8 \pm 0.6(100.0\%)$	$-14.88 \pm 0.47(100.0\%)$	$-14.88 \pm 0.47(100.0\%)$	$-14.869 \pm 0.52(100.0\%)$	$-14.949 \pm 0.36(100.0\%)$
$g02$	$-0.73865 \pm 0.01(100.0\%)$	$-0.74115 \pm 0.01(100.0\%)$	$-0.73865 \pm 0.01(100.0\%)$	$-0.73807 \pm 0.03(100.0\%)$	$-0.73865 \pm 0.01(100.0\%)$
$g03$	inf(0.0%)	inf(0.0%)	$-0.45446 \pm 0.14(100.0\%)$	inf(0.0%)	$\mathbf{-0.5746 \pm 0.18(100.0\%)}$
$g04$	$-30666.0 \pm 0.0(100.0\%)$	inf(0.0%)	$-30666.0 \pm 0.0(100.0\%)$	$-30665.0 \pm 0.05(100.0\%)$	$-30666.0 \pm 0.0(100.0\%)$
$g05$	inf(0.0%)	inf(0.0%)	inf(0.0%)	$\mathbf{5326.3 \pm 2.6e+02(54.0\%)}$	$\mathbf{5395.3 \pm 2.9e+02(100.0\%)}$
$g06$	$-6961.8 \pm 0.0(100.0\%)$	inf(0.0%)	$-6961.8 \pm 0.0(100.0\%)$	$-6961.8 \pm 0.02(100.0\%)$	$-6961.8 \pm 0.0(100.0\%)$
$g07$	$25.714 \pm 0.25(100.0\%)$	$25.82 \pm 0.35(96.0\%)$	inf(0.0%)	$\mathbf{25.697 \pm 0.31(100.0\%)}$	$124.75 \pm 60.0(100.0\%)$
$g08$	$-0.095825 \pm 0.0(100.0\%)$	$-0.095825 \pm 0.0(100.0\%)$	$-0.095825 \pm 0.0(100.0\%)$	$-0.095825 \pm 0.0(100.0\%)$	$-0.095825 \pm 0.0(100.0\%)$
$g09$	$680.64 \pm 0.01(100.0\%)$	$680.64 \pm 0.01(100.0\%)$	inf(0.0%)	$680.64 \pm 0.01(100.0\%)$	$740.4 \pm 71.0(100.0\%)$
$g10$	$7378.7 \pm 1.1e+02(36.0\%)$	inf(0.0%)	inf(0.0%)	$\mathbf{7651.9 \pm 1.7e+03(82.0\%)}$	$\mathbf{12383.0 \pm 2.5e+03(100.0\%)}$
$g11$	$0.76046 \pm 0.02(100.0\%)$	$0.74992 \pm 0.0(100.0\%)$	inf(0.0%)	$0.75018 \pm 0.0(100.0\%)$	$0.76206 \pm 0.02(100.0\%)$
$g12$	$-0.97803 \pm 0.0(100.0\%)$	$-0.97803 \pm 0.0(100.0\%)$	$-0.97803 \pm 0.0(100.0\%)$	$-0.97803 \pm 0.0(100.0\%)$	$-0.97803 \pm 0.0(100.0\%)$
$g13$	inf(0.0%)	inf(0.0%)	inf(0.0%)	$\mathbf{1.3465 \pm 1.5(40.0\%)}$	$\mathbf{0.77801 \pm 0.29(100.0\%)}$
$g14$	inf(0.0%)	inf(0.0%)	$-43.912 \pm 1.2(100.0\%)$	inf(0.0%)	$-43.525 \pm 1.4(100.0\%)$
$g15$	inf(0.0%)	$966.15 \pm 3.2(66.0\%)$	inf(0.0%)	$\mathbf{963.4 \pm 2.0(100.0\%)}$	$\mathbf{963.6 \pm 1.9(100.0\%)}$
$g16$	$-1.9052 \pm 0.0(100.0\%)$	$-1.9052 \pm 0.0(100.0\%)$	$-1.9052 \pm 0.0(100.0\%)$	$-1.9052 \pm 0.0(100.0\%)$	$-1.9052 \pm 0.0(100.0\%)$
$g17$	inf(0.0%)	inf(0.0%)	inf(0.0%)	$\mathbf{9125.0 \pm 1.4e+02(10.0\%)}$	$\mathbf{8991.5 \pm 1.0e+02(100.0\%)}$
$g18$	$-0.85146 \pm 0.04(76.0\%)$	$-0.78742 \pm 0.11(94.0\%)$	$-0.79976 \pm 0.09(100.0\%)$	$-0.81395 \pm 0.08(100.0\%)$	$-0.83094 \pm 0.07(100.0\%)$
$g19$	$35.305 \pm 1.3(100.0\%)$	$35.446 \pm 1.9(98.0\%)$	inf(0.0%)	$35.331 \pm 1.6(100.0\%)$	$709.84 \pm 2.0e+02(100.0\%)$
$g20$	inf(0.0%)	inf(0.0%)	inf(0.0%)	inf(0.0%)	inf(0.0%)
$g21$	inf(0.0%)	inf(0.0%)	inf(0.0%)	inf(0.0%)	inf(0.0%)
$g22$	inf(0.0%)	inf(0.0%)	inf(0.0%)	inf(0.0%)	inf(0.0%)
$g23$	inf(0.0%)	inf(0.0%)	$-68.268 \pm 42.0(12.0\%)$	inf(0.0%)	$\mathbf{123.17 \pm 1.8e+02(36.0\%)}$
$g24$	$-5.508 \pm 0.0(100.0\%)$	$-5.508 \pm 0.0(98.0\%)$	$-5.508 \pm 0.0(100.0\%)$	$-5.508 \pm 0.0(100.0\%)$	$-5.508 \pm 0.0(100.0\%)$

TABLE IV

MEAN OBJECTIVE VALUES AND STANDARD DEVIATION FOR THE FEASIBLE RUNS OUT OF 50 RUNS IN TOTAL USING THE GIVEN HOMOGENEOUS AND HETEROGENEOUS CONSTRAINT HANDLING METHODS. THE PROPORTION OF FEASIBLE RUNS IS GIVEN IN PARENTHESES. *inf* INDICATES THAT NO FEASIBLE SOLUTION WAS FOUND.

[4] C. A. Coello Coello. Theoretical and Numerical Constraint-Handling Techniques used with Evolutionary Algorithms: A Survey of the State of the Art. *Computer Methods in Applied Mechanics and Engineering*, 191(11–12):1245–1287, 2002.

[5] D. W. Coit, A. E. Smith, and D. M. Tate. Adaptive penalty methods for genetic optimization of constrained combinatorial problems. *INFORMS Journal on Computing*, 8:173–182, 1996.

[6] K. Deb. An Efficient Constraint Handling Method for Genetic Algorithms. *Computer Methods in Applied Mechanics and Engineering*, 186(2):311–338, 2000.

[7] A. B. Hadj-Alouane and J. C. Bean. A genetic algorithm for the multiple-choice integer program. *Operations Research*, 45(1):pp. 92–101, 1997.

[8] S. Helwig. *Particle Swarms for Constrained Optimization*. PhD thesis, University of Erlangen-Nuremberg, Germany, 2010. http://www.opus.ub.uni-erlangen.de/opus/volltexte/2010/1933/.

[9] X. Hu, R. C. Eberhart, and Y. Shi. Engineering optimization with particle swarm. In *Proc. 2003 IEEE Swarm Intelligence Symposium (SIS)*, pages 53–57, 2003.

[10] J. Joines and C. R. Houck. On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GA's. In *Proc. 1st IEEE Conference on Evolutionary Computation*, pages 579–584. IEEE Press, 1994.

[11] J. Kennedy and R. C. Eberhart. Particle swarm optimization. In *Proc. IEEE International Conference on Neural Networks*, volume 4, pages 1942–1948, 1995.

[12] J. Kennedy and R. C. Eberhart. A discrete binary version of the particle swarm algorithm. In *Proc. IEEE International Conference on Systems, Man and Cybernetics*, volume 5, pages 4104–4108, 1997.

[13] J. Kennedy and R. Mendes. Population structure and particle swarm performance. In *Proc. 2002 IEEE Congress on Evolutionaly Computation (CEC)*, pages 1671–1676, 2002.

[14] S. Koziel and Z. Michalewicz. A decoder-based evolutionary algorithm for constrained parameter optimization problems. *Evolutionary Computation*, 4:1–32, 1998.

[15] S. Koziel and Z. Michalewicz. Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. *Evolutionary Computation*, 7:19–44, 1999.

[16] J. J. Liang, T. P. Runarsson, E. Mezura-Montes, M. Clerc, P. N. Suganthan, C. A. Coello Coello, and K. Deb. Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. KanGAL Report 2006006, Nanyang Technological University, Singapore, 2006.

[17] R. Mendes. *Population Topologies and Their Influence in Particle Swarm*

	Super	Static×Super	Super×Static	Super×Dynamic	Super×Deb
$g01$	$-14.96 \pm 0.28(100.0\%)$	$-14.869 \pm 0.52(100.0\%)$	$-14.778 \pm 0.67(100.0\%)$	$-14.96 \pm 0.28(100.0\%)$	$-14.84 \pm 0.54(100.0\%)$
$g02$	$-0.73865 \pm 0.01(100.0\%)$	$-0.73865 \pm 0.01(100.0\%)$	$-0.73865 \pm 0.01(100.0\%)$	$-0.73713 \pm 0.03(100.0\%)$	$-0.73865 \pm 0.01(100.0\%)$
$g03$	$-0.48961 \pm 0.16(100.0\%)$	$\mathbf{-0.57032 \pm 0.19(100.0\%)}$	$-0.47844 \pm 0.18(100.0\%)$	$\inf(0.0\%)$	$\mathbf{-0.51133 \pm 0.17(100.0\%)}$
$g04$	$-30666.0 \pm 0.0(100.0\%)$	$-30666.0 \pm 0.0(100.0\%)$	$-30666.0 \pm 0.0(100.0\%)$	$-30665.0 \pm 0.04(100.0\%)$	$-30666.0 \pm 0.0(100.0\%)$
$g05$	$5244.1 \pm 1.8e+02(100.0\%)$	$5441.4 \pm 3.0e+02(100.0\%)$	$\inf(0.0\%)$	$5331.4 \pm 2.9e+02(62.0\%)$	$5258.6 \pm 2.0e+02(100.0\%)$
$g06$	$-6961.8 \pm 0.0(100.0\%)$	$-6961.8 \pm 0.0(100.0\%)$	$-6961.8 \pm 0.0(100.0\%)$	$-6961.8 \pm 0.01(100.0\%)$	$-6961.8 \pm 0.0(100.0\%)$
$g07$	$25.78 \pm 0.34(100.0\%)$	$\mathbf{25.76 \pm 0.37(100.0\%)}$	$\mathbf{25.746 \pm 0.29(100.0\%)}$	$\mathbf{25.737 \pm 0.3(100.0\%)}$	$105.54 \pm 59.0(100.0\%)$
$g08$	$-0.095825 \pm 0.0(100.0\%)$	$-0.095825 \pm 0.0(100.0\%)$	$-0.095825 \pm 0.0(100.0\%)$	$-0.095825 \pm 0.0(100.0\%)$	$-0.095825 \pm 0.0(100.0\%)$
$g09$	$680.64 \pm 0.01(100.0\%)$	$680.64 \pm 0.01(100.0\%)$	$680.64 \pm 0.0(100.0\%)$	$680.64 \pm 0.01(100.0\%)$	$728.47 \pm 51.0(100.0\%)$
$g10$	$7370.4 \pm 1.1e+02(100.0\%)$	$7382.3 \pm 1.1e+02(100.0\%)$	$\mathbf{7364.0 \pm 1.1e+02(100.0\%)}$	$\mathbf{7286.4 \pm 2.0e+02(100.0\%)}$	$10696.0 \pm 1.3e+03(100.0\%)$
$g11$	$0.75074 \pm 0.0(100.0\%)$	$\mathbf{0.75017 \pm 0.0(100.0\%)}$	$\mathbf{0.75039 \pm 0.0(100.0\%)}$	$\mathbf{0.75 \pm 0.0(100.0\%)}$	$0.75509 \pm 0.01(100.0\%)$
$g12$	$-0.97803 \pm 0.0(100.0\%)$	$-0.97803 \pm 0.0(100.0\%)$	$-0.97803 \pm 0.0(100.0\%)$	$-0.97803 \pm 0.0(100.0\%)$	$-0.97803 \pm 0.0(100.0\%)$
$g13$	$0.6488 \pm 0.25(100.0\%)$	$0.70529 \pm 0.23(100.0\%)$	$0.833 \pm 0.2(16.0\%)$	$0.91954 \pm 0.75(50.0\%)$	$0.75195 \pm 0.34(100.0\%)$
$g14$	$-43.982 \pm 1.1(100.0\%)$	$-43.621 \pm 1.4(100.0\%)$	$-42.836 \pm 1.8(38.0\%)$	$\inf(0.0\%)$	$\mathbf{-44.46 \pm 1.2(100.0\%)}$
$g15$	$962.98 \pm 1.5(100.0\%)$	$963.05 \pm 1.6(100.0\%)$	$963.1 \pm 0.75(8.0\%)$	$963.16 \pm 1.6(100.0\%)$	$\mathbf{962.71 \pm 1.4(100.0\%)}$
$g16$	$-1.9052 \pm 0.0(100.0\%)$	$-1.9052 \pm 0.0(100.0\%)$	$-1.9052 \pm 0.0(100.0\%)$	$-1.9052 \pm 0.0(100.0\%)$	$-1.9052 \pm 0.0(100.0\%)$
$g17$	$8926.2 \pm 31.0(100.0\%)$	$8955.7 \pm 71.0(100.0\%)$	$\inf(0.0\%)$	$9056.8 \pm 1.4e+02(18.0\%)$	$8952.1 \pm 80.0(100.0\%)$
$g18$	$-0.8357 \pm 0.07(100.0\%)$	$\mathbf{-0.84403 \pm 0.06(100.0\%)}$	$\mathbf{-0.86289 \pm 0.01(100.0\%)}$	$-0.82805 \pm 0.08(100.0\%)$	$-0.82603 \pm 0.07(100.0\%)$
$g19$	$35.505 \pm 1.6(100.0\%)$	$35.634 \pm 1.6(100.0\%)$	$35.538 \pm 1.6(100.0\%)$	$\mathbf{35.424 \pm 1.2(100.0\%)}$	$700.92 \pm 1.9e+02(100.0\%)$
$g20$	$\inf(0.0\%)$	$\inf(0.0\%)$	$\inf(0.0\%)$	$\inf(0.0\%)$	$\inf(0.0\%)$
$g21$	$251.35 \pm 38.0(100.0\%)$	$252.53 \pm 35.0(100.0\%)$	$231.83 \pm 0.0(2.0\%)$	$297.2 \pm 30.0(10.0\%)$	$283.27 \pm 61.0(78.0\%)$
$g22$	$289.01 \pm 52.0(50.0\%)$	$\inf(0.0\%)$	$\inf(0.0\%)$	$\inf(0.0\%)$	$\inf(0.0\%)$
$g23$	$8.6538 \pm 1.4e+02(100.0\%)$	$101.67 \pm 2.0e+02(98.0\%)$	$\inf(0.0\%)$	$161.0 \pm 1.5e+02(22.0\%)$	$86.961 \pm 2.1e+02(96.0\%)$
$g24$	$-5.508 \pm 0.0(100.0\%)$	$-5.508 \pm 0.0(100.0\%)$	$-5.508 \pm 0.0(100.0\%)$	$-5.508 \pm 0.0(100.0\%)$	$-5.508 \pm 0.0(100.0\%)$

TABLE V

MEAN OBJECTIVE VALUES AND STANDARD DEVIATION FOR THE FEASIBLE RUNS OUT OF 50 RUNS IN TOTAL USING THE CONSTRAINT HANDLING METHODS INVOLVING THE *super* STRATEGY FROM [25]. THE PROPORTION OF FEASIBLE RUNS IS GIVEN IN PARENTHESES. *inf* INDICATES THAT NO FEASIBLE SOLUTION WAS FOUND.

Performance. PhD thesis, Universidade do Minho, Portugal, 2004.

[18] Z. Michalewicz and C. Z. Janikow. GENOCOP: A genetic algorithm for numerical optimization problems with linear constraints. *Commun. ACM*, 1996. Article no. 175.

[19] Z. Michalewicz and M. Schoenauer. Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary Computation*, 4(1):1–32, 1996.

[20] S. Minton, M. D. Johnston, A. B. Philips, and P. Laird. Minimizing conflicts: A heuristic repair method for constraint-satisfaction and scheduling problems. *Artificial Intelligence*, 58:161–205, 1993.

[21] A. F. K. Morales and J. Gutiérrez-García. Penalty Function Methods for Constrained Optimization with Genetic Algorithms: A Statistical Analysis. In *Proc. 2nd Mexican International Conference on Artificial Intelligence*, pages 108–117. Springer, 2002.

[22] A. F. K. Morales and C. C. Quezada. A universal eclectic genetic algorithm for constrained optimization. In *Proc. 6th European Congress on Intelligent Techniques & Soft Computing (EUFIT)*, pages 518–522, 1998.

[23] R. Motwani, J. Naor, and P. Raghavan. Randomized approximation algorithms in combinatorial optimization. In D. S. Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*, pages 447–481. PWS, 1996.

[24] K. E. Parsopoulos and M. N. Vrahatis. *Particle Swarm Optimization and Intelligence: Advances and Applications*. Information Science Publishing, 2010.

[25] G. Toscano Pulido and C. A. Coello Coello. A constraint-handling mechanism for particle swarm optimization. In *Proc. 2004 IEEE Congress on Evolutionary Computation (CEC)*, volume 2, pages 1396–1403, 2004.

[26] B. W. Wah, Y. Chen, and A. Wan. Constrained global optimization by constraint partitioning and simulated annealing. In *Proc. IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 265–274, 2006.

[27] F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945.