Available online at www.sciencedirect.com
ScienceDirect
Applied Soft Computing 8 (2008) 337–349
Applied Soft Computing
www.elsevier.com/locate/asoc

Fitness inheritance in multiple objective evolutionary algorithms: A test bench and real-world evaluation

E.I. Ducheyne [a], B. De Baets [b,*], R.R. De Wulf [c]

[a] Veterinary Department, Institute of Tropical Medicine, Nationalestraat 155, 2000 Antwerpen, Belgium
[b] Department of Applied Mathematics, Biometrics and Process Control, Ghent University, Coupure Links 653, 9000 Gent, Belgium
[c] Laboratory of Forest Management and Spatial Information, Ghent University, Coupure Links 653, 9000 Gent, Belgium

Received 14 January 2005; received in revised form 1 February 2007; accepted 8 February 2007
Available online 28 February 2007

Abstract

In many real-world applications of evolutionary algorithms, the fitness of an individual has to be derived using complex models and time-consuming computations. Especially in the case of multiple objective optimisation problems, the time needed to evaluate these individuals increases exponentially with the number of objectives due to the 'curse of dimensionality' [J. Chen, D.E. Goldberg, S. Ho, K. Sastry, Fitness inheritance in multi-objective optimization, in: W.B. Langdon et al. (Eds.), GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference, July 9–13, Morgan Kaufmann Publishers, New York, 2002, pp. 319–326]. This in turn leads to a slower convergence of the evolutionary algorithms. It is not feasible to use time-consuming models with large population sizes unless the time to evaluate the objective functions is reduced. Fitness inheritance is an efficiency enhancement technique that was originally proposed by Smith et al. [R.E. Smith, B.A. Dike, S.A. Stegmann, Fitness inheritance in genetic algorithms, in: Proceedings of the 1995 ACM Symposium on Applied Computing, February 26–28, ACM, Nashville, TN, USA, 1995] to improve the performance of genetic algorithms. Sastry et al. [K. Sastry, D.E. Goldberg, M. Pelikan, Don't evaluate, inherit, in: L. Spector et al. (Eds.), GECCO 2001: Proceedings of the Genetic and Evolutionary Computation Conference, Morgan Kaufmann Publishers, San Francisco, 2001, pp. 551–558] and Chen et al. [J. Chen, D.E. Goldberg, S. Ho, K. Sastry, Fitness inheritance in multi-objective optimization, in: W.B. Langdon et al. (Eds.), GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference, July 9–13, Morgan Kaufmann Publishers, New York, 2002, pp. 319–326] have developed analytical models for fitness inheritance. In this paper, the usefulness of fitness inheritance for a set of popular and separable multiple objective test functions as well as a non-separable real-world problem is evaluated based on unary performance measures testing closeness to the Pareto-optimal front, uniform distribution along and extent of the obtained Pareto front. A statistical evaluation of the performance of an NSGA-II like algorithm on the basis of these unary performance measures suggests that especially for non-convex or non-continuous problems the use of fitness inheritance negatively affects the closeness to the Pareto-optimal front.

Keywords: Fitness inheritance; Multiple objective genetic algorithm; Real-world application

1. Theoretical foundations of fitness inheritance

1.1. Single objective fitness inheritance

Smith et al. [19] were the first to introduce the fitness inheritance technique. They point out that for some (and probably most) real-world optimisation problems, genetic algorithms (GA) cannot be applied because the cost of determining the fitness values for an entire population is too high. Earlier on, Grefenstette and Fitzpatrick [12] tried to reduce the evaluation time by partially evaluating each individual instead of completely evaluating it and by allowing the genetic algorithm to run for a larger number of generations. They concluded that it is more effective to evaluate fast noisy fitness functions for more generations than to evaluate the slow but exact functions for fewer generations. Smith et al. [19] attempted a similar procedure, but instead of evaluating parts of the individuals, they evaluated only parts of the population. In order to derive a fitness value for the offspring that is not evaluated, they proposed two methods: the *average inheritance*, where the offspring's fitness is calculated as the average value of the fitness of its parents, and the *proportional inheritance* where

* Corresponding author. Tel.: +32 9 264 59 41; fax: +32 9 264 62 20.
E-mail address: Bernard.DeBaets@ugent.be (B. De Baets).

1568-4946/$ – see front matter
doi:10.1016/j.asoc.2007.02.003

this average is weighed according to the similarity between the offspring and its parents.

Smith et al. [19] applied the schema theory to explain why the genetic algorithm performance is not disturbed by the noisy fitness function and based their calculations on two characteristics of genetic algorithms:

(1) A child can inherit schemata common to both parents and in that case, the schemata fitness values are correctly determined. The update of the genetic algorithm then reflects the average fitness of the common schemata in the individuals.
(2) A child can inherit schemata that are only present in one parent and this leads to an approximate fitness value of those schemata.

In the case of fitness inheritance, the fitness values of the schemata belonging to one parent only are either deemed constant in the case of average fitness inheritance, or these values are considered to be linearly related to the number of bits it inherited from its parents in the case of proportional fitness inheritance.

For the One Max problem, Smith et al. [19] compared the performance of the conventional genetic algorithm with that of the inheritance techniques. When all individuals were evaluated, the optimum (all 1s) is reliably reached after 10,200 evaluations. In the case of the inheritance approaches this was attained after only 2640 function evaluations. They argued that this might be caused by the simplicity of the One Max problem and applied the same techniques to an aircraft routing problem. For this real-world application, the length of the flying route had to be minimised, while the aircraft had to avoid detection by a threat along the route. Each time the aircraft was detected, a penalty was added to the objective function. Their preliminary results showed that the best results could be obtained if only one individual at each generation is evaluated, provided that elitism is applied. This showed that fitness inheritance had the potential to improve GA performance.

Even though this approach seemed promising, it took another six years before this work was continued. Sastry et al. [17] investigated the time to convergence, population sizing and the optimal proportion of inheritance for the One Max problem. The optimal proportion is the amount of inheritance that can be used in such a way that the number of function evaluations is minimised. They found that the time until population convergence is given by

$$t_{\text{conv}} = \frac{\pi}{2I}\sqrt{\frac{l}{1-p_i}}, \tag{1}$$

where I is the selection intensity, l the length of the chromosome and p_i is the fraction of the individuals that inherit their value. The population size n can be written as

$$n = -\frac{2^{k-1}\log(\psi)\sqrt{\pi}}{1-p_i^3}\sqrt{\sigma_f^2}, \tag{2}$$

where k is the size of the building block, ψ the failure rate or the rate that one accepts for not reaching the optimal value and σ_f^2 is the variance of the noisy fitness functions. This noise is caused by the incorrect fitness value for the inheritance individuals. Finally, p_i is once more the proportion of individuals that inherit fitness values. They also determined that the optimal proportion of inheritance p_i^* lies between 54 and 55.8%. This is considerably less than what Smith et al. [19] indicated: they calculated that for the One Max problem the proportion of inheritance could be raised up to 90% without loss of optimality. By building these analytical models, Sastry et al. [17] were the first to provide a strong theoretical basis for fitness inheritance.

1.2. Multiple objective fitness inheritance

Chen et al. [3] extended the analytical model provided by Sastry et al. [17] to multiple objective problems. They included an extra parameter M to account for the number of niches in the multiple objective problem. The problem for which the population sizing model and the time to convergence was derived was the bi-objective One Max problem. This problem is defined by

$$\text{Maximise} \quad \begin{cases} f_1(s,x_1) = l - d(s,x_1) \\ f_2(s,x_2) = l - d(s,x_2) \end{cases} \tag{3}$$

where s is the string to be evaluated, x_1 and x_2 the two fixed reference strings, l the string length and $d(s, x_i)$ is the Hamming distance between string s and string x_i. Chen et al. [3] used as reference string x_1 all ones and as reference string x_2 all ones except for the first four bits. Their model for convergence time is

$$t_{\text{conv}} = \frac{\pi}{2I}\sqrt{\frac{l}{1-p_i}}\sqrt{1+\frac{M-1}{l}}. \tag{4}$$

The population size n can be determined as follows

$$n = -\frac{2^{k-1}\log(\psi)M\sqrt{\pi}}{1-p_i^3}\sqrt{\sigma_f^2+\sigma_N^2}, \tag{5}$$

where σ_N^2 is the noise variance of the other niches. These models are very similar to those by Sastry et al. [17] and if $M = 1$ they reduce to the single objective models.

Both models were experimentally tested on the bi-objective One Max problem. It was found that when the inheritance proportion is smaller than 0.7, the results fit the predicted convergence and population-sizing model, but for large inheritance proportions the models were no longer valid.

Other than the preliminary results for the aircraft routing problem by Smith et al. [19], there has been no report of real-world applications that use fitness inheritance. The question arises whether this efficiency enhancement technique can cope with real-world applications.

To investigate this, both the average and proportional inheritance techniques will be tested on a test suite of functions provided by Zitzler [21] and Zitzler et al. [22], which is the most widely suite of benchmark problems employed (Huband et al. [14]).

2. Test functions and performance measures

2.1. Test functions

Zitzler [21] and Zitzler et al. [22] presented a set of six test functions to test whether multiple objective genetic algorithms can cope with specific characteristics (i.e. convexity, concavety and discontinuity). Based on preliminary results presented in Ducheyne et al. [7], where the outcome of fitness inheritance was visually inspected, these six functions where regrouped into three categories. In this paper three functions from the test suite are used because they represent three different categories: ZDT1, a convex function, ZDT2, a function with a non-convex Pareto front and ZDT3, a discontinuous function. Each of these functions has independent genes, thus it should be feasible to use fitness inheritance to speed up the optimisation process. Consider the following functions:

$$f(x_1) = x_1,$$
$$g(x_2, \ldots, x_n) = 1 + \frac{9 \cdot \left(\sum_{i=2}^{n} x_i\right)}{n-1},$$

where $n = 30$ and $x_i \in [0, 1]$.

(1) ZDT1 has a convex Pareto-optimal front: $f_1 = f$, $g_1 = g$ and

$$h_1(f_1, g_1) = 1 - \sqrt{\frac{f_1}{g_1}}. \tag{6}$$

The Pareto-optimal front is formed when g_1 equals 1.

(2) ZDT2 is the non-convex counterpart of the first test function: $f_2 = \mathrm{f}$, $g_2 = g$ and

$$h_2(f_2, g_2) = 1 - \left(\frac{f_2}{g_2}\right)^2. \tag{7}$$

The Pareto-optimal front is formed when g_2 equals 1.

(3) ZDT3 tests whether a genetic algorithm is able to cope with discreteness in the Pareto-optimal front: $f_3 = \mathrm{f}$, $g_3 = g$ and

$$h_3(f_3, g_3) = 1 - \sqrt{\frac{f_3}{g_3}} - \left(\frac{f_3}{g_3}\right) \sin(10\pi f_3). \tag{8}$$

The Pareto-optimal front is formed when g_3 equals 1. The sine function introduces discontinuity in the Pareto-optimal front but not in the objective space.

In each of the above cases, the goal is to minimise the functions f_i, g_i, h_i simultaneously. The other three functions from the test suite also fall under one of the above categories. Because separable test functions are not representative for real-world problems (Bäck and Michalewicz [2]), a well-studied non-separable real world forest management optimisation problem (Ducheyne et al. [8–10]) was also used as test problem.

2.2. Performance measures

For the analysis of the fitness inheritance, the three previously described test functions are used. The Pareto fronts achieved by the evolutionary algorithm without fitness inheritance, as well as the Pareto fronts for the two genetic algorithms with average or proportional fitness inheritance, will be presented. The Pareto-optimal front is drawn for comparison purposes. The comparison of the outcome of different algorithms is not an easy task. Various unary and binary performance measures have been proposed in the literature. The use of unary performance measures to determine whether one algorithm is better than another is inappropriate as was shown by Zitzler et al. [24]. However, in this paper we want to investigate whether the use of fitness inheritance has an influence on specific performance characteristics which cannot be analysed based on n-ary performance measures. Because of this, we will apply commonly used unary performance measures in order to evaluate the following characteristics of a multiple objective genetic algorithm:

(1) closeness to the Pareto-optimal front;
(2) the distribution of solutions along the obtained Pareto front;
(3) the extent of the obtained Pareto front.

2.2.1. Testing closeness

Van Veldhuizen and Lamont [20] proposed two measures: the error ratio and the generational distance. The *error ratio* is simply calculated as the ratio of the number of solutions not on the Pareto-optimal front to the population size:

$$M_1 = \frac{\sum_{i=1}^{n} e_i}{n}, \tag{9}$$

where $e_i = 0$ if individual i is on the Pareto-optimal front P^* and $e_i = 1$ if it is not. This measure is bounded by 0 (all solutions are on the Pareto-optimal front) and 1 (none of the solutions are on the Pareto-optimal front). One of the main drawbacks of this measure is that it can only indicate whether there are any solutions on the Pareto-optimal front. If for two algorithms none of the solutions are on this front, the error ratio cannot distinguish between them [4]. Therefore, Deb redefines the error conditions as follows [4]: if the minimum distance between an individual i and the Pareto-optimal front P^* is larger than a threshold δ, then the individual is counted as an error. If a suitable threshold is used, then the δ-error ratio can give an indication about the proportion of individuals within a distance δ of the Pareto-optimal front [4].

Another measure proposed by Van Veldhuizen and Lamont [20] is the generational distance. The *generational distance* is a value representing how far the current front is from the optimal front. The value is defined as

$$M_2 = \frac{\sqrt{\sum_{i=1}^{N} n d_i^2}}{n}, \tag{10}$$

where n is the number of individuals in the current front and d_i is the distance between each of the individuals and the nearest individual on the Pareto-optimal front. This measure can also be used as a measure of fluctuation between several repetitions. Deb [4] proposes to use the variance in the distance values to test the robustness of an algorithm. Both measures defined by Van Veldhuizen and Lamont [20] are scaling dependent.

2.2.2. Testing spread among the non-dominated points

Schott [18] introduced a spacing measure based on a variance-like measure. For the spacing measure the variance between the distances of each solution on the Pareto front and the mean distance along the Pareto front is calculated. The *spacing measure* is defined by

$$M_3 = \frac{1}{n-1}\sum_{i=1}^{n}(d_i - \bar{d})^2. \tag{11}$$

2.2.3. Testing the extent of the obtained Pareto-front

Deb and Jain [6] extended the spacing measure because the it does not take into account the maximum spread between the most extreme solutions. The *spread measure* is defined by

$$M_4 = \frac{\sum_{m=1}^{M} d_m^{\mathrm{e}} + \sum_{i=1}^{Q}(d_i - \bar{d})}{\sum_{m=1}^{M} d_m^{\mathrm{e}} Q\bar{d}}, \tag{12}$$

where Q denotes the number of solutions in the Pareto front, d_i the Euclidean distance of solution i to the closest solution j in the Pareto front, $\bar{d}$ the average of these distances and d_m^{e} is the distance between the extreme solutions of the Pareto and Pareto-optimal fronts according to objective dimension m.

If the spacing and spread measures are averaged over a number of repetitions, then a low mean value indicates evenly spaced solutions, while a high mean value indicates that the solutions are not very well distributed along the front. Knowles and Corne [16] indicate that these measures can only be used in conjunction with other measures. In that case it provides information about the vector distributions. An advantage of these measures is their low computational cost.

2.2.4. Combining spread, extent and closeness

Another measure introduced by Zitzler [21] is the hypervolume, which calculates the hypervolume enclosed by a solution set A and a reference point. It computes the area of the search space dominated by this solution set. In many aspects this measure is a very good one [16,23] but it has one caveat: the size of the dominated space is easily influenced by the reference point, and care has to be taken before any decisive conclusions based on this measure can be made. A disadvantage is the larger computational overhead, but While et al. [14] have developped a faster algorithm to compute the hypervolume.

2.3. Performance analysis

The generational distance, error ratio, spacing, spread and hypervolume measures are used to compare the different approaches. These measures are calculated every 100 function evaluations for the non-inheritance approach and every 50 function evaluations for the inheritance techniques. For all three algorithms this is done at every generation. The difference in means of the above measures over time is determined by a One Way ANOVA test if the data is normally distributed and homoscedastic, otherwise a non-parametric Kruskal–Wallis test is applied. All tests are done at a significance level of 95%.

3. Solving the test functions with fitness inheritance

3.1. Parameter settings

The experiments were performed using a NSGA-II like genetic algorithm with binary tournament selection, one-point crossover with crossover probability of 0.8 and a uniform mutation rate of 0.01. The population size was set to 100 and the number of generations was set to 200. These settings are the same as in [21], only differing in the number of generations. The encoding of the decision vector was also the same as in [21]: an individual is represented as a bit vector where each parameter x_i is represented by 30 bits. The crowding distance assignment procedure was used for sharing. The fitness assignment was proposed by Deb et al. [5] and equals the number of solutions that dominate the current solution. All experiments were repeated 10 times. The reported optimal proportion of 54% is used to test the performance of the genetic algorithms. The maximum number of fitness evaluations for the inheritance approach should be the same as for the non-inheritance approach, therefore the number of generations is doubled to 400 generations to allow for a fair comparison.

3.2. Convex functions

3.2.1. Visual comparison

The standard genetic algorithm is capable of finding a Pareto front very close to the optimal Pareto front for the first convex test function (Fig. 1). Both inheritance strategies approximate the Pareto-optimal front well. If there is no inheritance, the variance between the different points is low; on the other hand, the points from the proportional inheritance approach are much more scattered. The solutions from the two inheritance approaches are also much more concentrated near the point $(f_1, h_1) = (0, 1)$.

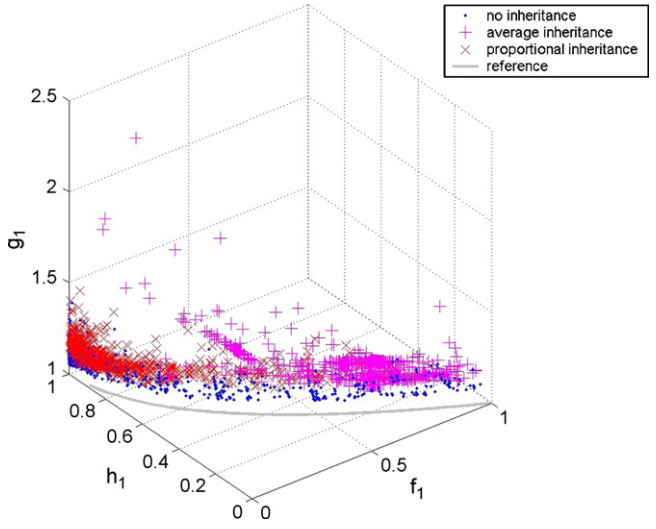

Fig. 1. The overall Pareto front for test function 1. On the x-axis $F_1 = f_1$, on the y-axis $F_2 = h_1$ and on the z-axis $F_3 = g_1$.

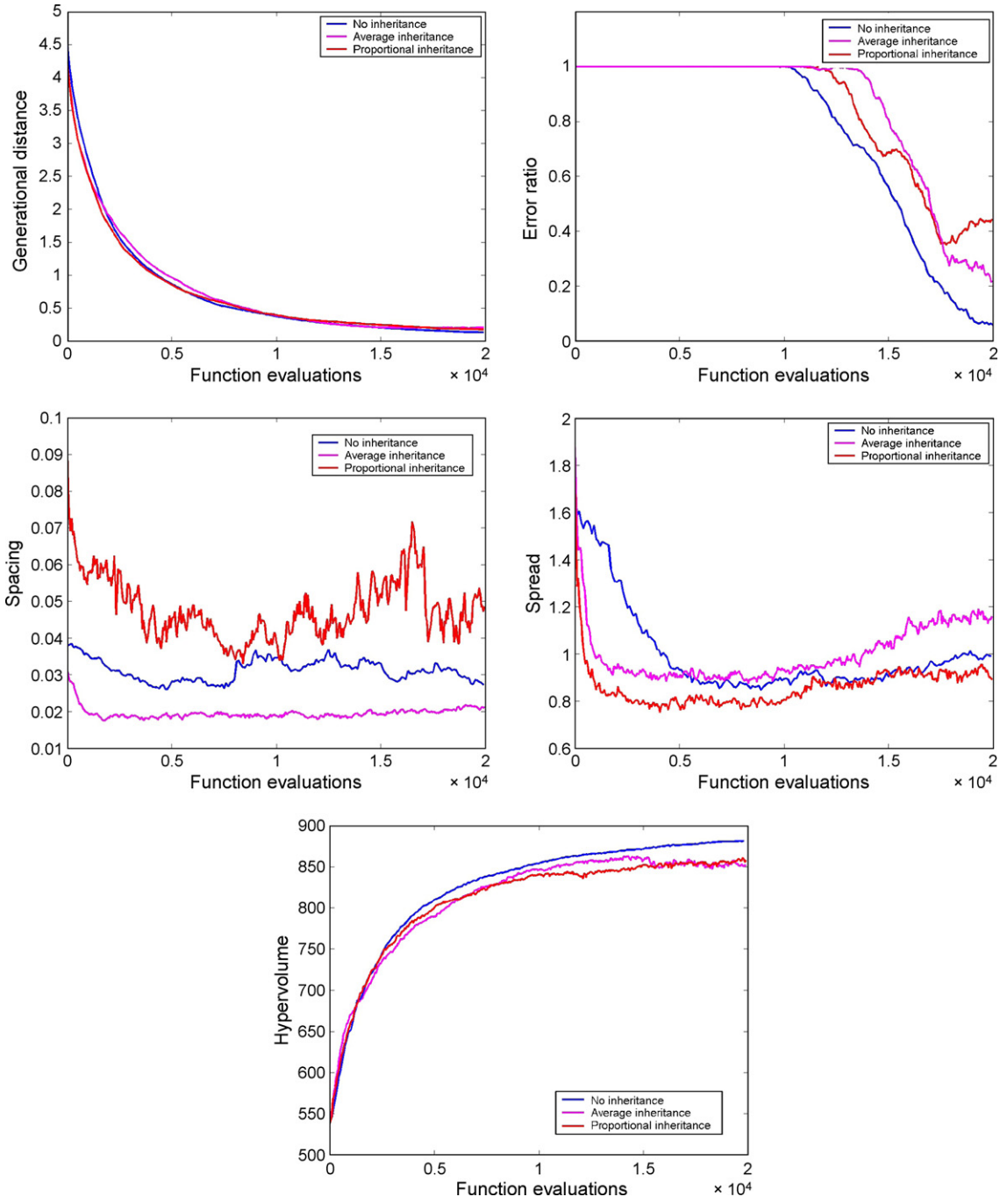

Fig. 2. Evolution of mean generational distance, mean error ratio, mean spacing, mean spread and mean hypervolume over the number of function evaluations for the first test function (number of repetitions = 10).

3.2.2. Generational distance

The evolution of the mean generational distance computed over 10 repetitions over the function evaluations for the standard genetic algorithm is very similar to that of the two inheritance techniques (Fig. 2). In all three cases, the decrease in generational distance is initially strong, but becomes smaller as the population converges. The data was statistically analysed using a One Way ANOVA test because on all occasions it was normally distributed and homoscedastic. All p-values for the relevant tests of normality (Shapiro–Wilk), homoscedasticity (Levene's test) and One Way ANOVA are listed in Table 1. Before 1800 function evaluations, the generational distance is significantly lower for the inheritance techniques than for the regular genetic algorithm ($p = 0.000$). From then on, there is no difference ($p > 0.05$) until 16,800 function evaluations. After that, the generational distance for the non-inheritance approach is significantly lower than for the proportional inheritance approach but not than the average inheritance technique ($p = 0.037$) according to Tukey's posthoc-test.

Table 1
p-Values for normality, homoscedasticity and One Way ANOVA for different numbers of function evaluations for generational distance for the first test function (non, no inheritance; average, average inheritance; proportional, proportional inheritance)

Evaluations	Group	Shapiro–Wilk	Levene	One Way ANOVA
100	Non	0.971	0.776	0.000
	Average	0.662		
	Proportional	0.380		
1,800	Non	0.688	0.776	0.067
	Average	0.885		
	Proportional	0.695		
16,800	Non	0.482	0.919	0.052
	Average	0.984		
	Proportional	0.493		
16,900	Non	0.470	0.902	0.037
	Average	0.910		
	Proportional	0.407		

Table 3
p-Values for normality, homoscedasticity and Kruskal–Wallis for different numbers of function evaluations for spread for the first test function (number of repetitions = 10), for 20,000 function evaluations One Way ANOVA is used (non, no inheritance; average, average inheritance; proportional, proportional inheritance)

Evaluations	Group	Shapiro–Wilk	Levene	Kruskal–Wallis
3,200	Non	0.200	0.010	0.000
	Average	0.193		
	Proportional	0.644		
6,000	Non	0.390	0.216	0.002
	Average	0.017		
	Proportional	0.636		
10,000	Non	0.442	0.004	0.000
	Average	0.455		
	Proportional	0.583		
20,000	Non	0.428	0.058	0.000
	Average	0.483		
	Proportional	0.713		

3.2.3. Error ratio

The previous findings are confirmed by the error ratio: initially it is 1 for all algorithms, but after 10,000 function evaluations the error ratio declines quickly. The error ratio of the inheritance techniques is higher than that of the standard genetic algorithm, especially near the end of the evolution (Fig. 2). Apparently the convergence towards the front is hampered by the fitness inheritance. The effect of the inheritance on the error ratio was also investigated statistically: for each 100 function evaluations, the Kruskal–Wallis test was performed because the data was not normally distributed in all cases, and because the data was not homoscedastic. According to this test, there is no significant difference present between the three approaches, but near the end of the evolution, the test statistic p is close to 0.05. This indicates that if the number of function evaluations increases even more, the inheritance techniques might perform worse than the regular technique.

3.2.4. Spacing and spread

Spacing and spread on the other hand are not so much influenced by the inheritance techniques: in all cases the crowding distance operator ensures that the spacing and spread remain more or less equal over the complete evolution. The effect of the inheritance techniques was again tested using a Kruskal–Wallis test as once more the data was not normally distributed nor homoscedastic (Table 2). Initially the spacing measure for the non-inheritance approach is significantly lower than for the other two approaches ($0.1 \leq p \leq 0.2$), but after 17,500 function evaluations there is no longer a difference between non-inheritance and average inheritance (Fig. 2). In the last phase, it is inconclusive whether there is a difference or not: $p = 0.047 \approx 0.05$.

The Kruskal–Wallis test for the spread measure clearly shows that in all cases there are significant differences. For all cases $p = 0$ even though this is not clear from Fig. 2. All p-values for the relevant tests of normality (Shapiro–Wilk), homoscedasticity (Levene's test) and One Way ANOVA and Kruskal–Wallis are listed in Table 3. Initially all groups are significantly different from each other ($p = 0.000$), and the standard genetic algorithm performs the worst in terms of spread. At 6000 function evaluations, they have more or less the same performance but still the Kruskal–Wallist test indicates significant differences ($p = 0.002$). Even at the end of the evolution there is still a significant difference between the proportional inheritance and no-inheritance strategy ($p = 0.000$) according to Tukey's posthoc test.

It seems that the amount of spacing and spread is highly determined by the initial spacing in the population. Especially spacing does not change much over the course of the genetic algorithm. Spread is lowered somewhat more than spacing but

Table 2
p-Values for normality, homoscedasticity and Kruskal–Wallis for different numbers of function evaluations for spacing for the first test function (number of repetitions = 10) (non, no inheritance; average, average inheritance; proportional, proportional inheritance)

Evaluations	Group	Shapiro–Wilk	Levene	Kruskal–Wallis
17,500	Non	0.377	0.481	0.422
	Average	0.010		
	Proportional	0.010		

Table 4
p-Values for normality, homoscedasticity and One Way ANOVA for different numbers of function evaluations for hypervolume for the first test function (number of repetitions = 10) (non, no inheritance; average, average inheritance; proportional, proportional inheritance)

Evaluations	Group	Shapiro–Wilk	Levene	One Way ANOVA
4,000	Non	0.960	0.180	0.000
	Average	0.482		
	Proportional	0.773		
16,000	Non	0.930	0.719	0.002
	Average	0.482		
	Proportional	0.773		

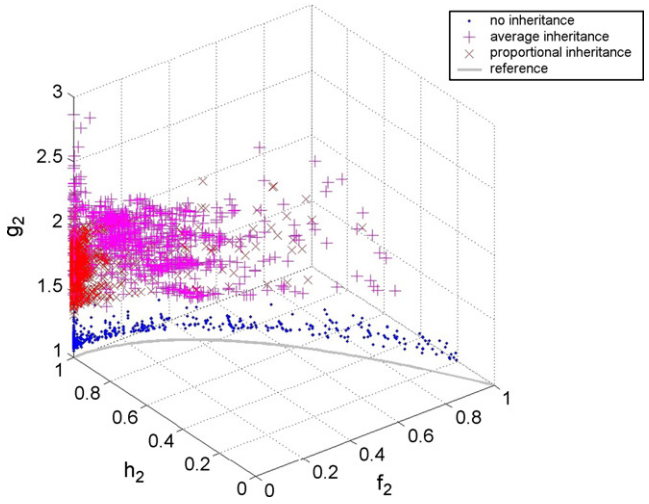

Fig. 3. The overall Pareto front for test function 2. On the x-axis $F_1 = f_2$, on the y-axis $F_2 = h_2$ and on the z-axis $F_3 = g_2$.

still remains much the same. This might indicate that the crowding distance measure is more a 'diversity preserving' measure than a 'diversity stimulating' measure.

3.2.5. Hypervolume

The inheritance techniques show a higher error ratio and this is confirmed by the hypervolume measure: it is lower for both inheritance techniques than for the standard genetic algorithm (Fig. 2). All p-values for the relevant tests of normality (Shapiro–Wilk), homoscedasticity (Levene's test) and One Way ANOVA are listed in Table 4. Up until 4000 function evaluations this difference is not significant. Between 4000 and 16,000 there is a difference between the non-inheritance and the average inheritance approach according to Tukey's posthoc-test. Beyond this point this posthoc-test indicates a significant difference between the non-inheritance approach on the one hand and the two inheritance strategies on the other hand.

3.3. Functions with a non-convex Pareto front

3.3.1. Visual comparison

The algorithms based on fitness inheritance are not suitable for solving optimisation problems with a non-convex Pareto front (Fig. 3). The points found by the inheritance approaches are much further away from the Pareto-optimal front. Furthermore, there are many points near the origin of the x-axis. This phenomenon was also observed for the first test function.

3.3.2. Generational distance

The generational distance is higher for both inheritance techniques than for the non-inheritance approach and this is true for the complete evolution (Fig. 4). Between the two inheritance techniques, however, there is a visual difference. The data was statistically analysed using a One Way ANOVA. All p-values for the relevant tests of normality (Shapiro–Wilk), homoscedasticity (Levene's test) and One Way ANOVA are listed in Table 5. At 600 function evaluations, the ANOVA test statistic shows that there are no significant differences between the three algorithms ($p = 0.21$). At 700 function evaluations this changes and there is no conclusion possible as $p = 0.050$. At 800 evaluations however $p = 0.018$. Tukey's posthoc-test indicates that there is a significant difference between the non-inheritance and the average inheritance approach, but is inconclusive for the proportional approach. This remains the same until 1600 function evaluations. At that point, Tukey's posthoc-test shows that there is a significant difference between the non-inheritance approach on the one hand and both inheritance techniques on the other hand.

3.3.3. Error ratio

The error ratio of the two inheritance techniques is much worse than for the non-inheritance approach. Whereas the error ratio drops after 10,000 evaluations for the regular genetic algorithm, the error ratio for the two inheritance approaches stays 1 for the complete duration of the genetic algorithm. As their error ratio remains constant, no statistical analysis is possible. Still, from Fig. 4 follows clearly that the inheritance techniques do not perform well in terms of error ratio, and that they fail to reach the Pareto-optimal front for a function with a non-convex Pareto front.

3.3.4. Spacing and spread

Spacing and spread are again not influenced by the inheritance techniques. The crowding distance operator ensures once more that the spacing and spread remains more or less equal over the complete evolution (Fig. 4).

The spacing of the non-inheritance approach is less than that of the two inheritance approaches, and the spacing of the average inheritance is better than that of the proportional inheritance technique. Before 8000 function evaluations the difference between the spacing is significant only between non-inheritance and proportional inheritance. Beyond this point, there is a difference between the non-inheritance approach and the two inheritance approaches (Table 6).

Looking at spread, initially the non-inheritance approach performs worse than the two other approaches. This difference is not significant with respect to the average inheritance approach, but is significant with respect to the proportional inheritance approach until 6000 function evaluations ($p = 0.000$) (Table 7). After 6000 function evaluations, there is no longer a difference between the non-inheritance and inheritance approaches but there is a significant difference between the two inheritance procedures: according to Tukey's posthoc test, average inheritance has a significantly higher spread than proportional inheritance ($p = 0.013$).

3.3.5. Hypervolume

The hypervolume measure again confirms the findings of the error ratio and the generational distance: this value is lower for both inheritance techniques than for the standard genetic algorithm (Fig. 4). Up until 1400 function evaluations, this difference is not significant between the three groups ($p = 0.054$) (Table 8). Between 1400 and 1800 function evaluations, the non-inheritance approach is only significantly

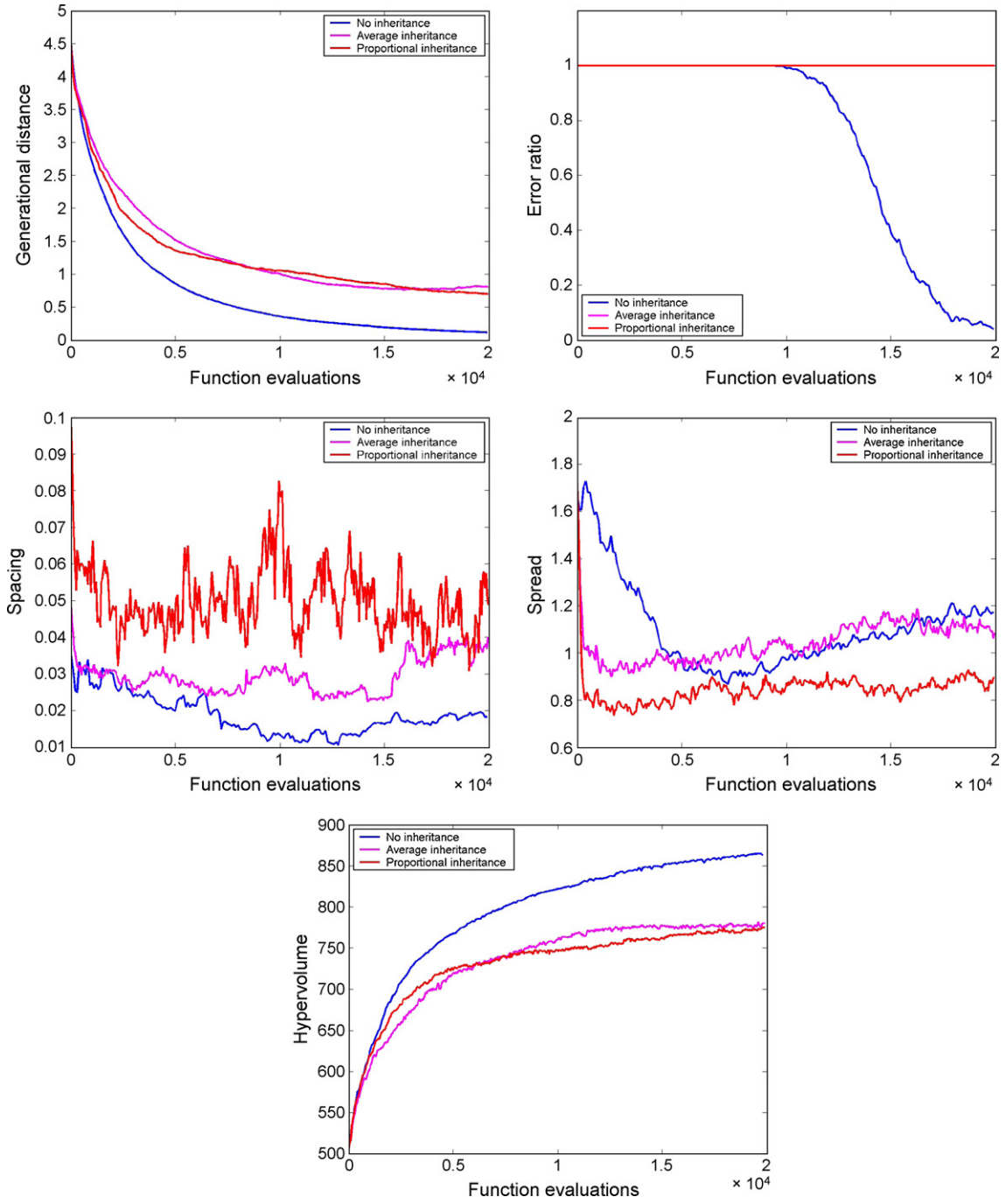

Fig. 4. Evolution of mean generational distance, mean error ratio, mean spacing, mean spread and mean hypervolume over the number of function evaluations for the second test function (number of repetitions = 10).

different ($p = 0.000$) from the average approach, but later on Tukey's posthoc-test indicates that all groups are significantly different from each other.

3.4. Discontinuous functions

3.4.1. Visual comparison

The inheritance techniques are unable to solve the discontinuous problem (Fig. 5). Both inheritance techniques yield the same result but their Pareto fronts approximate the Pareto-optimal front in a linear way. This is particularly pronounced in the case of the average fitness inheritance. The solutions are also unevenly distributed along the fronts. Apparently, the noise resulting from the linear interpolation of the fitness values of the offspring results in a high disturbance of the genetic algorithm.

3.4.2. Generational distance

The generational distance is similar for all approaches for the complete evolution (Fig. 6). Between the two inheritance techniques, however, there is no visual difference. The data was statistically analysed using a One Way ANOVA. All p-values

Table 5
p-Values for normality, homoscedasticity and One Way ANOVA for different numbers of function evaluations for generational distance for the second test function (number of repetitions = 10) (non, no inheritance; average, average inheritance; proportional, proportional inheritance)

Evaluations	Group	Shapiro–Wilk	Levene	One Way ANOVA
600	Non	0.344	0.098	0.210
	Average	0.491		
	Proportional	0.511		
700	Non	0.668	0.111	0.050
	Average	0.761		
	Proportional	0.487		
800	Non	0.779	0.183	0.018
	Average	0.989		
	Proportional	0.595		
1600	Non	0.135	0.497	0.001
	Average	0.499		
	Proportional	0.357		

for the relevant tests of normality (Shapiro–Wilk), homoscedasticity (Levene's test) and One Way ANOVA are listed in Table 9. Early in the evolution, there is not much difference between the approaches, but according to Tukey's posthoc-test this difference becomes significant among all groups after 10,000 function evaluations.

3.4.3. Error ratio

The error ratio of the two inheritance techniques is much worse than for the non-inheritance approach (Fig. 6). Whereas the error ratio drops after 10,000 evaluations for the regular genetic algorithm, the error ratio for the average inheritance approach stays 1 for the complete duration of the genetic algorithm. The error ratio of the proportional inheritance technique does decline after 16,000 function evaluations but much less than in the non-inheritance case.

3.4.4. Spacing and spread

As was the case with the two other functions, the spacing and spread are not affected by the inheritance. The spacing of the average inheritance is again much lower than that of the others. From Fig. 6 can be concluded that once more the points from the average inheritance are well distributed, but that proportional inheritance shows a higher degree of scatter.

This is confirmed by the spread measure: the spread measures for the three approaches are similar and the values for

Table 6
p-Values for normality, homoscedasticity and Kruskal–Wallis for different numbers of function evaluations for spacing for the second test function (number of repetitions = 10) (non, no inheritance; average, average inheritance; proportional, proportional inheritance)

Evaluations	Group	Shapiro–Wilk	Levene	Kruskal–Wallis
4,800	Non	0.010	0.151	0.021
	Average	0.655		
	Proportional	0.438		
8,000	Non	0.010	0.288	0.003
	Average	0.203		
	Proportional	0.073		
16,000	Non	0.010	0.632	0.012
	Average	0.648		
	Proportional	0.048		

Table 7
p-Values for normality, homoscedasticity and One Way ANOVA for different numbers of function evaluations for spread for the second test function (non, no inheritance; average, average inheritance; proportional, proportional inheritance)

Evaluations	Group	Shapiro–Wilk	Levene	One Way ANOVA
4,000	Non	0.247	0.422	0.000
	Average	0.527		
	Proportional	0.728		
6,000	Non	0.175	0.645	0.013
	Average	0.468		
	Proportional	0.696		
10,000	Non	0.764	0.626	0.0000
	Average	0.424		
	Proportional	0.448		

Table 8
p-Values for normality, homoscedasticity and One Way ANOVA for different numbers of function evaluations for hypervolume for the second test function (number of repetitions = 10) (non, no inheritance; average, average inheritance; proportional, proportional inheritance)

Evaluations	Group	Shapiro–Wilk	Levene	One Way ANOVA
1400	Non	0.692	0.269	0.054
	Average	0.349		
	Proportional	0.871		
1800	Non	0.951	0.324	0.000
	Average	0.378		
	Proportional	0.560		

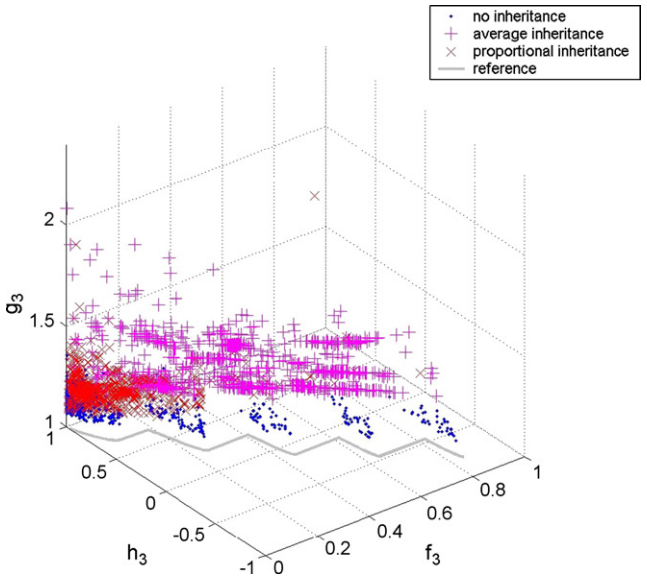

Fig. 5. The overall Pareto front for test function 3. On the *x*-axis $F_1 = f_2$, on the *y*-axis $F_2 = h_2$ and on the *z*-axis $F_3 = g_2$.

Table 9
p-Values for normality, homoscedasticity and One Way ANOVA for different numbers of function evaluations for generational distance for the third test function (number of repetitions = 10) (non, no inheritance; average, average inheritance; proportional, proportional inheritance)

Evaluations	Group	Shapiro–Wilk	Levene	One Way ANOVA
10,000	Non; average; proportional	0.348; 0.828; 0.516	0.846	0.000

the average technique are now in the range of the other two techniques (Fig. 6). However, because the distance to the extreme points is taken into account, this compensates for the low values of the spacing measure.

3.4.5. Hypervolume

Finally the hypervolume measure confirms the findings for the error ratio and the generational distance (Fig. 6): this value is lower for the inheritance techniques than for the standard

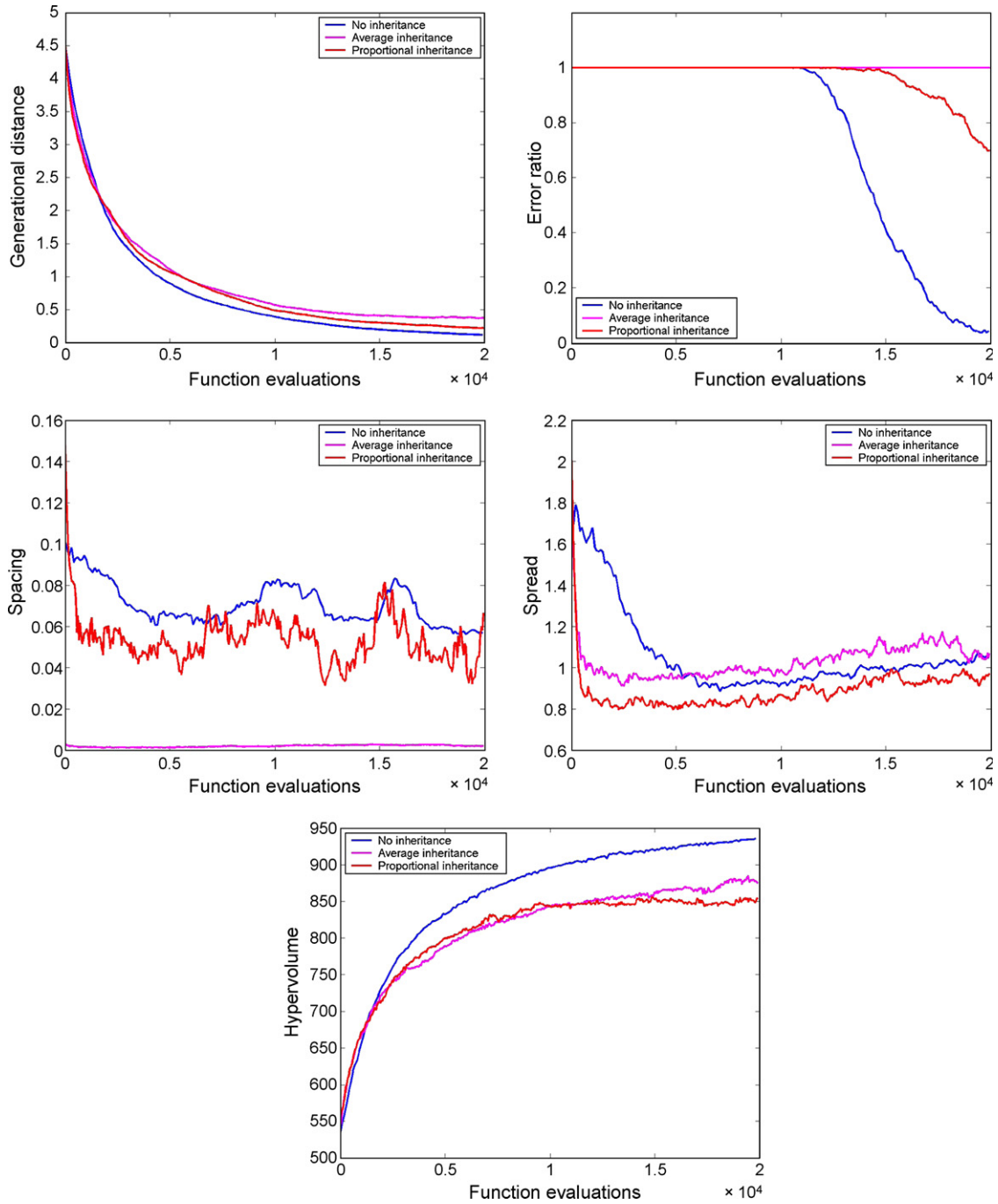

Fig. 6. Evolution of mean generational distance, mean error ratio, mean spacing, mean spread and mean hypervolume over the number of function evaluations for the third test function (number of repetitions = 10).

Table 10
p-Values for normality, homoscedasticity and One Way ANOVA for different numbers of function evaluations for hypervolume for the third test function (number of repetitions = 10) (non, no inheritance; average, average inheritance; proportional, proportional inheritance)

Evaluations	Group	Shapiro–Wilk	Levene	One Way ANOVA
2700	Non	0.485	0.068	0.040
	Average	0.491		
	Proportional	0.444		

genetic algorithm. Up until 2700 function evaluations this difference is not significant between the three groups (Table 10). After 2700 there is a difference between the non-inheritance and the two inheritance approaches, but these two do not differ from each other until the very end of the evolution.

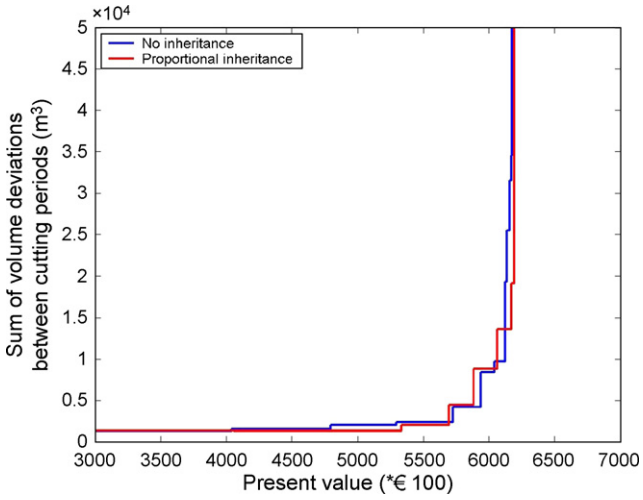

Fig. 7. Attainment surfaces for the harvest scheduling problem for non-inheritance and proportional inheritance approaches.

4. Fitness inheritance for harvest scheduling

4.1. Introduction

As a final case study, fitness inheritance is used to speed up the optimisation process for a bi-objective harvest scheduling problem. Forest managers need to schedule management treatments over a planning horizon. Often the objective is to maximise the net present value, while minimising at the same time the deviations between the different cutting periods. The net present value can be calculated as

$$f = \frac{V_t}{(1+i)^t}, \tag{13}$$

where V_t is the revenue obtained at period t and i is the discount rate. The timber production is calculated from production forecast tables produced by the British Forestry Commission [13]. This produced timber is then multiplied by current prices [1]. As the production tables of the trees do not change over time, it is possible to combine these two sources in order to obtain the current value for the timber retrieved from the forest.

The second objective for this harvest scheduling problem is the minimisation of the deviations in timber volume per cutting period. This is needed to ensure an even flow of timber volume towards the wood processing industry over the complete planning horizon. According to the formulation of Johnson and Scheurman [15] for a Model I harvest scheduling problem, this can be written as

$$g = \sum_{j=1}^{M} V_j - \bar{V}, \tag{14}$$

where M is the number of time periods, V_j the total volume (m^3) cut in period j and $\bar{V}$ is the average volume cut over all cutting periods.

This planning horizon is 80 years for this case study and a forest management unit can be scheduled for harvesting every 10 years. As a management unit can receive only one set of management actions for the complete duration of the planning horizon, a chromosome with the same number of genes as the number of management units is a sufficient representation for this problem. After analysis it followed that an integer representation is the best coding for this type of forest management problem. As this problem is convex, fitness inheritance should be a feasible approach. A confounding factor might be the inseparability of the problem. Ducheyne et al. [8] showed that the building blocks had a size up to 5.

4.2. Parameter settings

The population size is set to 100, the number of generations without fitness inheritance to 200, and with proportional inheritance to 400. Average inheritance was not tested because from the test functions followed that its performance was either similar to or worse than that of proportional inheritance. One-point crossover is used with a probability of 0.8 and uniform mutation with a probability of 0.01. Binary tournament selection was used and the crowding distance operator ensured sharing.

4.3. Results and discussion

From Fig. 7 follows that after the same number of function evaluations, the attainment surface from the inheritance approach equals that of the non-inheritance approach. This is confirmed by calculating the hypervolume measure. The data is normally distributed ($p = 0.99 > 0.05$) and homoscedastic ($p = 0.685$). From the Student's t-test test statistic follows that there is no significant difference between the two groups ($p = 0.209$).

The performance of the inheritance technique is thus very similar to that of the non-inheritance approach. It must be noted that still the same number of function evaluations are needed to ensure this performance and that there is no real gain of using the fitness inheritance.

5. Conclusions

Between the two fitness inheritance techniques there is little difference in performance. The generational distance, error ratio and hypervolume are not always significantly different, and in most cases very close to the values of the non-inheritance approach. Spacing and spread are even better for the average fitness inheritance than for the standard genetic algorithm. Overall fitness inheritance can speed up the optimisation process for convex functions while maintaining the same performance as the regular genetic algorithm.

It can be concluded that for functions with a non-convex Pareto front, after a few fitness evaluations, the inheritance techniques converge slower to the Pareto-optimal front than the non-inheritance approach. Maintaining spread or spacing is again not affected by the inheritance. This means that for functions with a non-convex Pareto front, inheritance techniques are less suitable to use.

As was the case for functions with a non-convex Pareto front, the inheritance techniques are not useful for discontinuous functions. Their performance in terms of generational distance, error ratio and hypervolume is significantly worse than for the non-inheritance approach over the complete evolution. Again, spacing and spread are not affected by the fitness inheritance.

Fitness inheritance efficiency enhancement techniques can be used in order to reduce the number of fitness evaluations provided that the Pareto front is convex and continuous. If the surface is not convex, the fitness inheritance strategies fail to reach the Pareto-optimal front. As to the inheritance strategy, it is safer to choose the proportional approach, because in most cases the proportional inheritance performs better in terms of generational distance, error ratio and hypervolume.

If real-world practitioners want to use fitness inheritance, it is advisable to check beforehand what the nature (convex, non-convex, ...) of the Pareto front is. This can be achieved by solving the problem with a classical multiple objective GA for a low number of generations and then switch to fitness inheritance techniques if there is sufficient indication that the Pareto front is convex and continuous. They should also consider other techniques for speeding up the optimisation process. These might include local fitness recalculation, where the fitness value of the individuals is updated from the value from the parents by recalculating only where the children differ from the parents.

The performance of the inheritance approach is similar to that of the standard genetic algorithm for the real-world application. However, this should be relativised because in reality the same number of function evaluations are necessary to obtain the same Pareto front. Nevertheless despite the fact that this problem is non-separable, fitness inheritance had no negative impact on the performance.

Other techniques such as meta-modelling or response surface approximation may be better suited to solve this kind of problems because they can approximate the objective function non-linearly [11].

References

[1] Anon., Gemiddelde Prijzen van Hout op Stam, Houthandel en nijverheid, pp. 5–5, 2000.
[2] T. Bäck, Z. Michalewicz, Test landscapes, in: T. Back, D. Fogel, Z. Michalewicz (Eds.), Handbook of Evolutionary Computation, Oxford University Press, Oxford, 1997, pp. 14–20.
[3] J. Chen, D.E. Goldberg, S. Ho, K. Sastry, Fitness inheritance in multi-objective optimization, in: W.B. Langdon, et al. (Eds.), GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference, July 9–13, Morgan Kaufmann Publishers, New York, 2002 , pp. 319–326.
[4] K. Deb, Multi-Objective Optimization Using Evolutionary Algorithms, Wiley and Sons, Chichester, UK, 2001.
[5] K. Deb, S. Agrawal, A. Pratab, T. Meyarivan, A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II, KanGAL Report 200001, Indian Institute of Technology, Kanpur, India, 2000.
[6] K. Deb, S. Jain, Running Performance Metrics for Evolutionary Multi-objective Optimization, Technical Report 200004, Kanpur Genetic Algorithms Laboratory, 2000.
[7] E. Ducheyne, B. De Baets, R.R. De Wulf, Is fitness inheritance useful for real-world applications? Lect. Notes Comput. Sci. 2632 (2003) 31–42.
[8] E. Ducheyne, B. De Baets, R.R. De Wulf, Probabilistic models for linkage learning in forest management, in: Y. Jin (Ed.), Knowledge Incorporation in Evolutionary Computation, Springer, Berlin, 2005, pp. 177–195.
[9] E. Ducheyne, R.R. De Wulf, B. De Baets, Single versus multiple objective genetic algorithms for solving the even-flow forest management problem, Forest Ecol. Manage. 201 (2–3) (2004) 259–273.
[10] E. Ducheyne, R.R. De Wulf, B. De Baets, A spatial approach to forest-management optimization: linking GIS and multiple objective genetic algorithms, Int. J. GIS 20 (8) (2006) 917–928.
[11] M. Emmerich, A. Giotis, M. Özdemir, T. Bäck, K. Giannakoglu, Metamodel-assisted evolution strategies, in: J.J. Merelo Guervós, et al. (Eds.), Parallel Problem Solving From NaturePPSN VII, Springer, Berlin, 2002, pp. 361–370.
[12] J.J. Grefenstette, J.M. Fitzpatrick, Genetic Search with Approximate Function Evaluations, in: J.J. Grefenstette (Ed.), Proceedings of an International Conference on Genetic Algorithms and Their Applications, Lawrence Erlbaum, Hillsdale, USA, 1985, pp. 112–120.
[13] G.J. Hamilton, J.M. Christie, Forest management tables, in: Forestry Commission Booklet No. 34., Her Majesty's Stationery Office, London, UK, 1971.
[14] L. While, P. Hingston, L. Barone, S. Huband, A faster algorithm for calculating hypervolume, IEEE Trans. Evolut. Comput. 10 (1) (2006) 29–38.
[15] K.N. Johnson, H.L. Scheurman, Techniques for prescribing optimal timber harvest and investment under different objectives—discussion and synthesis, Forest Sci. Monogr. 18 (31) (1977).
[16] J. Knowles, D. Corne, On metrics for comparing nondominated sets, Proceedings of the 2002 Congress on Evolutionary Computation Conference, IEEE, 2002, pp. 711–716.
[17] K. Sastry, D.E. Goldberg, M. Pelikan, Don't evaluate, inherit, in: L. Spector, et al. (Eds.), GECCO 2001: Proceedings of the Genetic and Evolutionary Computation Conference, Morgan Kaufmann Publishers, San Francisco, 2001, pp. 551–558.
[18] J.R. Schott, Fault tolerant design using single and multi-criteria genetic algorithms, Master's Thesis, Massachusetss Institute of Technology, 1995.
[19] R.E. Smith, B.A. Dike, S.A. Stegmann, Fitness inheritance in genetic algorithms, in: Proceedings of the 1995 ACM Symposium on Applied Computing, February 26–28, ACM, Nashville, TN, USA, 1995.
[20] D.A. Van Veldhuizen, G.B. Lamont, Multiobjective evolutionary algorithm test suites, in: J. Carroll, et al. (Eds.), Proceedings of the 1999 ACM Symposium on Applied Computing, ACM, 1999, pp. 351–357.
[21] E. Zitzler, Evolutionary algorithms for multiobjective optimization: methods and applications, PhD Thesis, Institut für Technische Informatik und Kommunikation-snetze, 1999.

[22] E. Zitzler, K. Deb, L. Thiele, Comparison of multiobjective evolutionary algorithms: empirical results, Evolut. Comput. 8 (2) (2000) 173–195.
[23] E. Zitzler, M. Laumanns, L. Thiele, C.M. Foneseca, V. Grunert da Fonseca, Why quality assessment of multiobjective optimizers is difficult, in: W.B. Langdon, et al. (Eds.), GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference, July 9–13, Morgan Kaufmann Publishers, New York, 2002, pp. 666–674.
[24] E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, V. Grunert da Fonseca, Performance assessment of multiobjective optimisers: an analysis and review, IEEE Trans. Evolut. Comput. 7 (2) (2003) 117–132.