On the Assessment of Multiobjective Approaches to the Adaptive Distributed Database Management Problem

Joshua D. Knowles, David W. Corne and Martin J. Oates

School of Computer Science, Cybernetics and Electronic Engineering
University of Reading, Reading RG6 6AY, UK.
Tel: +44 (0) 118 931 8983; Fax: +44 (0) 118 975 1774
Email: j.d.knowles,d.w.corne@reading.ac.uk

Abstract. In this paper we assess the performance of three modern multiobjective evolutionary algorithms on a real-world optimization problem related to the management of distributed databases. The algorithms assessed are the Strength Pareto Evolutionary Algorithm (SPEA), the Pareto Archived Evolution Strategy (PAES), and M-PAES, which is a Memetic Algorithm based variant of PAES. The performance of these algorithms is compared using two distinct and sophisticated multiobjective-performance comparison techniques, and extensions to these comparison techniques are proposed. The information provided by the different performance assessment techniques is compared, and we find that, to some extent, the ranking of algorithm performance alters according to the comparison metric; however, it is possible to understand these differences in terms of the complex nature of multiobjective comparisons.

1 Introduction

In real-world applications, obtaining the complete set of Pareto optimal solutions for a multiobjective problem may be impossible to attain, and instead we seek a 'good' approximation to this set, which may in fact contain *no* true Pareto optima. Measuring the quality of approximations such as these is a problematic area but a variety of methods have been put forward (e.g. [1, 3, 5, 8–12]), and most have been put to use in establishing differences in algorithm performance. However, further investigation of these methods is much needed because, as yet, no single method can determine and present in a concise form, unequivocal information about both the statistical and geometric properties of an algorithm's approximations to the Pareto set.

In this paper, a real-world telecommunications problem is considered under two slightly different multiobjective formulations. In both cases, we wish to make no *a priori* judgments about the preferences of the telecommunications company with regard to how they might view the QoS delivered to customers as a function of the objectives considered. Three modern, and competitively-matched multiobjective optimization algorithms are to be tested on a suite of different versions of the problem representing a number of different scenarios. Thus, the aim is to judge which algorithm obtains the 'best' sets of solutions on this suite of problems, using only the weak Pareto assumption. Rather than just viewing the results from the single perspective of just one statistical method of judging performance, we instead consider and extend several methods, and observe the differences and similarities in the information they provide.

The three algorithms we consider are all modern solutions to the problem of Pareto optimization. All three of them are elitist approaches, and they each maintain a list of (a limited number of) the nondominated solutions that they find. The first algorithm, the Strength Pareto Evolutionary Algorithm (SPEA) of Zitzler and Thiele [11, 12], has been widely tested. The second algorithm is (1+1)-PAES, originally put forward by us as a simple baseline algorithm [2]. Its performance has also been tested elsewhere on several problems [3]. Third, is an extension of the basic (1+1)-PAES algorithm; a memetic approach called M-PAES. Its performance was shown to be competitive with SPEA on a suite of multiobjective 0/1 knapsack problems [4].

The remainder of this paper is organised as follows. In the next section, the Adaptive Distributed Database Management Problem (ADDMP) is defined in general, and then the particular choice of scenarios tackled here is explained. Section 3 provides details of the experimental methods used, including algorithm parameter settings, and the performance assessment techniques employed. In this section, extensions to previous methods of assessing performance are included. The results of the experiments are presented and discussed in Section 4. Finally, in the concluding section, a summary of the findings, and ideas for future work are given.

2 The Adaptive Distributed Database Management Problem

The Adaptive Distributed Database Management problem (ADDMP) is a problem faced by distributed database service providers (DDSPs), such as video-on-demand, genome databanks, and so forth. Oates and Corne [7] gives a detailed description, and C source code for the evaluation function can be found via the first author's website[1]. Here, we provide basic details of the ADDMP, aimed at conveying an understanding of its multiobjective nature.

A DDSP needs to regularly ensure that database users (clients) are receiving adequate quality of service (QoS). Indeed, clients' subscription to the database may involve guarantees from the DDSP of distinct levels of QoS, perhaps varying with subscription cost. A key factor in QoS is the delay (or response time) experienced by a client for a typical database query. In maximizing QoS, the DDSP aims to minimize the delay for each client. However, since copies of all or parts of the database exist on several perhaps globally distributed servers, this minimization must occur in the context of load balancing. That is, we may

[1] http://www.reading.ac.uk/~ssr97jdk

be able to minimize the delays experienced by certain clients by routing their queries to the fastest server which contains the required data; however, the extra load on this server will degrade the delays. So, the optimal solution will involve a careful balancing of clients across servers.

The ADDMP is hence the problem of finding the best client/server connection configuration, given a particular scenario which specifies details of the underlying communications network, server speeds, and access rates for each client. What counts as 'best' depends on many things, but a single-objective QoS measure will typically involve combining the *worst* client delay with the mean or median delays. However, such QoS measures are growing increasingly inadequate as distributed database service provision becomes more widespread and complex as regards the range of service guarantees on offer. For example, consider two potential solutions to a 5-client ADDMP in which the vectors of client delays (in milliseconds) are, respectively: Solution 1 (155, 130, 140, 140, 140), Solution 2 (350, 80, 90, 90, 90).

In a single-objective approach, which of solution 1 or 2 is preferred depends very much on the relative weightings given to the worst and mean (or median) components. It is hence complex, and perhaps impossible, to derive 'correct' relative weightings for these components, especially considering the widely different kinds of ADDMP scenarios which exist.

A multiobjective approach therefore seems more sensible and flexible. Client 1, for example, may have paid for a QoS guarantee which indicates that their delay will always be below 200 ms. Client 2, on the other hand, may have been given a guarantee that their delay would be always within 20% of the median delay level at any snapshot in time. With varied sets of factors like this, the task of an optimizer addressing an ADDMP would be to quickly produce a good and diverse spread of solution configurations, leaving it to a later decision process to then choose from these on the basis of the various QoS guarantees in operation for the clients currently using the service.

The problem we address in this paper is therefore that of quickly providing a good set of diverse ADDMP configurations, from which a second decision-making process can then choose the best according to prevailing QoS issues.

ADDMP Variants

ADDMP instances can occur in great variety. The numbers of clients and servers can range typically between 2 and 20, and the number of clients between 10 and several thousand. Access patterns can vary equally dramatically. E.g., access to share price and similar financial databases may be very frequent with constantly changing global activity, and hence re-optimization of client/server access configurations may need to occur every few minutes. In other scenarios, involving a small number of clients, re-optimization may only need to occur every few hours.

A key part of the datafile defining an ADDMP instance is an array of client access rates. Over time, we can expect these to vary, leading to different ADDMP instances requiring re-optimization to redistribute the load according to current usage.

In this paper we look at scenarios involving 10, 20, and 40 client/servers, and in each case we consider 5 separate problems which reflect possible changes in access patterns over time. Thereby, we are comparing the quality of SPEA, PAES, and M-PAES on the ADDMP over a wide but representative range of potential instances. We are interested particularly in ADDMPs which need constant, and hence fast, re-optimization. I.e., results must arrive quickly. Hence, in increasing order of the problem sizes, the maximum allowed number of evaluations is 500, 2000 and 5000.

We consider both 2-objective and 3-objective versions of each problem. In the 2-objective version, the objectives are the worst delay figure and the median delay figure. In the 3-objective version, the objectives are the 90% delay figure (i.e. 90% of clients will have a better delay figure than this), the 80% delay figure, and the median figure.

3 Parameter control and performance assessment

The problems of comparing and assessing the performance of multiobjective optimizers fall broadly into two categories; controlling the parameters of the various algorithms, and actually measuring the performance. Space restrictions preclude proper discussion relating how parameters were controlled, but the ranges of values used are shown in Fig. 1.

algorithm	p_c	crossover type	p_m	mutate type	internal pop.	external pop.	*l_fails*	*l_opt*	*cr_trials*	l
M-PAES	N/A	**uniform**	**1/L**	**flip**	5–20	80–95	1–20	2–100	**5**	5 / 3
SPEA	**0.8**	**uniform**	**1/L**	**flip**	5–20	80–95	N/A	N/A	N/A	N/A
1+1-PAES	N/A	N/A	**1/L**	**flip**	**1**	**99**	N/A	N/A	N/A	5 / 3

Fig. 1. Parameter settings for the three algorithms. Bold face indicates a fixed value in all experiments. The ranges of values used for the free parameters is shown. These were investigated on an *ad hoc* basis to provide 'best' performance. The two values shown for l, the number of bisection levels in the adaptive grid algorithm [4], refer to the values for the two and three objective problems, respectively. Please refer to [4] for a description of the parameters used in M-PAES.

The quality of a set of nondominated vectors can be assessed in several distinctly different ways, leading to many possible metrics. Our preferences (at least in this study) are towards methods that do not require knowledge of the true Pareto front, such as Generational Distance or Error Ratio [9], because these are often not available. Nor do we consider cardinal measures, e.g. the Coverage metric [11], and again Error Ratio, to be very satisfactory because they can give extremely misleading information (see [3]). Instead, metrics based on measuring the position of the discovered attainment surface (the boundary between dominated and nondominated regions) [1], or the size of the dominated region itself [5, 11], seem preferable because they conform more closely to the ideals of Pareto optimization. The specific metrics we use are described next.

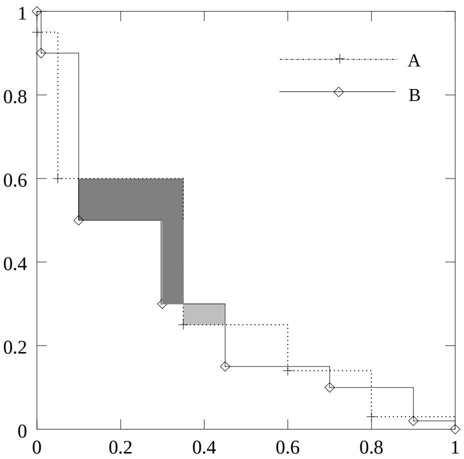

Fig. 2. A plot of the objective space of a 2-objective minimization problem. There are two attainment surfaces defined by two sets, A and B, of solution vectors. The combined dominated region $S(A \cup B)$ can be calculated easily. By subtracting the dominated region of A, $S(A)$, from $S(A \cup B)$, the combined size of the regions dominated by B but not A (e.g. dark shaded region), $S_{B \setminus A}$, can be calculated. Similarly, the combined size of regions dominated by A but not B (e.g. light shaded region), $S_{A \setminus B}$, can be calculated by subtracting $S(B)$ from $S(A \cup B)$.

In [1], a means of combining the information from several runs of an algorithm was put forward. To date, this has been our favoured means of performance assessment, and we have extended the approach to cater for comparison of multiple algorithms. As a whole, we refer to the method as *attainment surface sampling*. In brief, the method works by taking samples of the attainment surfaces defined by the discovered solution sets from independent runs of an algorithm or algorithms. The samples allow us to calculate and plot the best, median and worst attainment surfaces from a collection of runs. This information is not easy to analyse, however, except when there are only two objectives, so that surface plots are easy to interpret. Fortunately, statistical comparison of algorithms is also strongly catered for by these techniques; because each sampling of the attainment surfaces provides a univariate distribution, standard non-parametric statistics can be applied.

We use two statistical methods for comparing algorithms, one for comparing a pair of algorithms only, and another for comparing n algorithms at once. When comparing algorithms we must have a collection of solution sets from independent runs of each algorithm. If we adopt the phrase 'A beats B' at a particular sampled region of the Pareto front if in that region the distribution of attainment surfaces from multiple runs of algorithm A is better than that from multiple runs of algorithm B at a statistically significant level, according to some non-parametric test, then we can define our two measures: In the case of comparing just two algorithms denoted by A and B, the statistics returned are a pair of numbers, $\langle a, b \rangle$, where a indicates the percentage of the front on which A beats B, and where b indicates the percentage of the front on which B beats A. When comparing $n > 2$ algorithms, $n(n-1)$ pair-wise comparisons are made as above, but with the sampling of the surfaces always carried out on the full extent of the combined discovered Pareto front. Two statistics can then be defined for each algorithm. The algorithm's *unbeaten* statistic is the percentage of the front on which no other algorithm beats it, and it's *beats all* statistic is the percentage of the front on which it beats all of the other algorithms. For the univariate statistical tests that the above metrics depend upon, we use the Mann-Whitney U test [6] at the 95% confidence level. A fuller explanation of our implementation and criticisms of these techniques are given in [3].

Our other measures are based on the size of the dominated region, $S(C_\alpha)$, defined as the union of regions dominated by each individual solution vector in the set, C_α, of nondominated vectors found during optimization run α, within some bounding rectangular polytope, as in [5]. However, we do not size the bounding polytope based on the location of an ideal point, but just ensure that it contains all of the nondominated vectors. (Note: Both methods of normalization induce rather arbitrary scalings of the objectives that are hard to fully justify in a multiobjective context.) We propose using the size of the dominated region in three new ways, with the aim of combining the information contained in several runs of an algorithm, or to provide more useful comparative information. In the following, we also consider a collection C of n sets C_α, $\alpha \in \{1, ..., n\}$, of the points found in one optimization run of an algorithm.
Our first measure is of the total discovered region, found over a set of n runs:

$$S_\tau = S(\bigcup_{\alpha=1}^{n} C_\alpha) \tag{1}$$

The second measure combines the n runs by finding the size of the median attainment surface:

$$S_\mu = S(\mathcal{MS}(C)) \tag{2}$$

where $\mathcal{MS}(C)$ is the median attainment surface of the collection of solution sets, C, obtained using attainment surface sampling. Finally, we propose a measure for directly comparing two collections A and B of approximations, each obtained from n independent runs. The measure is based on the notion of the coverage difference of two sets [10] defined by:

$$S_{A \setminus B} = S(A \cup B) - S(B) \tag{3}$$

The meaning of the coverage difference is illustrated in Figure 2. This measure is useful when comparing two algorithms. Taking the two complements together, the degree to which one set of points covers the other can be inferred. For example, if the pair of complements $S_{A \setminus B}$ and $S_{B \setminus A}$ have the values 0 and 0.05 respectively then one can say that the points in B completely cover the points in A, and also that the points in A dominate 95% of the region dominated by B. Now, for a collection of runs from two algorithms, A and B, we propose calculating the median value of the coverage differences.

4 Results

In the following, all results are based on thirty independent runs of each algorithm on each problem instance. The first set of results (Fig. 3) was obtained using attainment surface sampling and our n-algorithm comparison metric. These

ADDMP instance	statistic	2-objective M-PAES	2-objective SPEA	2-objective 1+1-PAES	3-objective M-PAES	3-objective SPEA	3-objective 1+1-PAES
10-1	*unbeaten*	93.8	83.1	99.2	98.0	81.3	100
	beats all	0.8	0	5.0	0	0	2.0
10-2	*unbeaten*	100	99.8	94.9	68.5	98.2	100
	beats all	0	0	0	0	0	1.8
10-3	*unbeaten*	100	98.7	95.2	77.3	78.4	100
	beats all	0	0	0	0	0	8.3
10-4	*unbeaten*	100	98.9	100	64.3	51.8	100
	beats all	0	0	0	0	0	34.1
10-5	*unbeaten*	99.8	49.7	100	35.3	54.9	100
	beats all	0	0	0.2	0	0	16.3
20-1	*unbeaten*	100	48.7	100	95.8	24.1	100
	beats all	0	0	0	0	0	3.0
20-2	*unbeaten*	65.0	67.2	100	63.7	63.7	99.8
	beats all	0	0	32.8	0	0	36.3
20-3	*unbeaten*	100	0	100	98.9	67.0	100
	beats all	0	0	0	0	0	1.1
20-4	*unbeaten*	52.1	0.5	100	95.2	41.0	100
	beats all	0	0	47.9	0	0	4.8
20-5	*unbeaten*	49.9	50.0	100	23.8	24.7	100
	beats all	0	0	50.0	0	0	74.1
40-1	*unbeaten*	92.0	15.6	99.9	77.4	31.0	70.8
	beats all	0.1	0	8.0	9.9	0	22.6
40-2	*unbeaten*	69.3	7.9	93.4	76.4	26.2	87.6
	beats all	4.6	0	30.7	0.5	0	23.6
40-3	*unbeaten*	68.7	10.3	93.8	71.7	15.0	77.4
	beats all	4.2	0	31.3	10.9	0	28.3
40-4	*unbeaten*	100	11.2	99.5	78.2	0.1	93.1
	beats all	0	0	0	6.9	0	21.8
40-5	*unbeaten*	68.2	12.9	94.7	69.1	57.7	73.0
	beats all	4.9	0	31.8	0	0	30.9

Fig. 3. The *unbeaten* and *beats all* statistics for the combined space inhabited by the solutions found. Two forms of the problem were investigated: The 2-objective case, where the median response time and the worst response time are minimized. And the 3-objective case, where the median response time, the response time bettered by 80% of requests, and the response time bettered by 90% of requests are minimized. The different problems are labelled by the number of nodes and the number of the scenario. E.g. The third scenario of the twenty node problem is labelled 20-3.

results, taken on their own, seem to indicate that (1+1)-PAES is consistently difficult to beat, whereas SPEA and M-PAES are closely matched but not as consistently good as PAES. Results obtained from our other metrics, a key subset of which are presented below, are not always in agreement, however.

ADDMP instance	2-objective M-PAES	2-objective SPEA	2-objective 1+1-PAES	3-objective M-PAES	3-objective SPEA	3-objective 1+1-PAES
10-3	0.33803	0.33776	0.33810	0.2703528	0.2703444	0.2703528
20-3	0.33580	0.34616	0.34070	0.1322440	0.1322490	0.1322490
40-3	0.27758	0.30131	0.30210	0.0498326	0.0498368	0.0498400

Fig. 4. The values of the total dominated region, S_τ, for three problem instances.

ADDMP instance	2-objective M-PAES	2-objective SPEA	2-objective 1+1-PAES	3-objective M-PAES	3-objective SPEA	3-objective 1+1-PAES
10-3	0.1551	0.1590	0.1482	0.09570	0.0867	0.0960
40-1	0.342969	0.345659	0.345649	0.024767	0.026677	0.024202

Fig. 5. The values of the Median Attainment Region, S_μ, for two problem instances.

ADDMP instance	obj-ectives	A= M-PAES $S_{A\setminus B}$	B= 1+1-PAES $S_{B\setminus A}$	A= SPEA $S_{A\setminus B}$	B= 1+1-PAES $S_{B\setminus A}$	A= SPEA $S_{A\setminus B}$	B= M-PAES $S_{B\setminus A}$
sc10-1	2	0.0327	0.0124	0.0384	0.0134	0.0353	0.0216
sc10-1	3	0.0678	0.0	0.0161	0.0	0.0002	0.0266
sc20-4	2	0.0187	0.0161	0.0286	0.0153	0.0248	0.0118
sc20-4	3	0.0026	0.0	0.0239	0.0	0.0084	0.0009

Fig. 6. The median values of the coverage differences of alternate pairs of algorithms on two problem instances.

The results of calculating S_τ for three of the problems are shown in Fig. 4. First, notice that the values for the three algorithms are very close and that a large number of figures are significant. Nonetheless, the results here still provide extra information about the distribution of solutions found by the algorithms over multiple runs. On the first two of the 3-objective cases, two algorithms generate *exactly* the same total dominated region. In all probability this must indicate that the total set of non-dominated solutions found in each case is exactly the same. Interestingly, the total dominated region measure favours SPEA over M-PAES, although (1+1)-PAES is superior overall. Using our *unbeaten* and *beats all* statistics, M-PAES is ranked ahead of SPEA, with (1+1)-PAES in first place. This disagreement must indicate that SPEA tends to generate different solutions on different runs more often than M-PAES. Still, the total dominated region preserves the position of (1+1)-PAES as the most consistent algorithm, it winning on three of the six measurements, and being beaten only once.

In the next set of results presented (Fig. 5) the median attainment surface is first calculated for each algorithm. The size of the region dominated is then measured. Once again, using this measure alone could lead to different conclusions than if using it in conjunction with other measures. For example, on the two-objective version of problem 10-3, the rank order reported by this measure S_μ, the S_τ measure, and our *unbeaten* statistic are all different. Clearly the algorithms perform at very similar levels on this problem, but which is best? On the three objective version of the same problem, M-PAES and (1+1)-PAES are very similar with regard to S_μ, whereas M-PAES is a poor third when considering the whole distribution of attainment surfaces, using the Mann-Whitney U test.

Finally, the coverage differences, $S_{A\setminus B}$ and $S_{B\setminus A}$ were calculated for pairs of the algorithms, for each of the thirty runs. The median values of these differences are presented in Fig. 6. These results exhibit a high degree of agreement with the equivalent results in Fig. 3. According to the former, the attainment surface generated by (1+1)-PAES completely dominates the attainment surface found by the other two algorithms on at least 50% of the runs, on the three-objective versions of the two problems.

5 Conclusion

In this paper we have compared the performance of three multiobjective algorithms with respect to a suite of real-world problems related to the management of distributed databases. Several test metrics were employed to measure and compare algorithm performance over collections of solution sets found from several (30) runs. Three extensions to a metric based on the size of the dominated space were presented and used. The results show that with such closely-matched algorithms, it is a very difficult matter to select the one that performs best. In fact, from the results presented, (1+1)-PAES seems to be the best performer all-round. However, results given by the different metrics indicated that the rank order of algorithms was certainly not independent of the test metric. This shows that it is most important to use a number of different metrics, when comparing algorithms. Then, where a rank order is inconsistent across different metrics, the extra information provided can help to understand exactly in what way one algorithm's approximations differ from another's. Actually, more detailed analysis of the full results reveals that SPEA was most consistent at providing an even distribution of solutions along the front, whereas (1+1)-PAES more often found very strong compromise solutions.

The metrics we have proposed could also be extended further. For example, testing the coverage differences over a collection of runs, could be carried out by using a statistical test such as Kolmogorov-Smirnov, rather than the median. In addition, we are currently developing a method based on the S measure, that can calculate and present information about the individual regions where one attainment surface dominates another.

Acknowledgments

The authors thank the anonymous reviewers for their helpful comments. The first author would also like to express his gratitude to BT Labs plc. for sponsorship of his PhD.

References

1. C. M. Fonseca and P. J. Fleming. On the Performance Assessment and Comparison of Stochastic Multiobjective Optimizers. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature—PPSN IV*, Lecture Notes in Computer Science, pages 584–593. Springer-Verlag, Berlin, Germany, September 1996.
2. J. D. Knowles and D. W. Corne. The Pareto Archived Evolution Strategy: A New Baseline Algorithm for Multiobjective Optimisation. In *1999 Congress on Evolutionary Computation*, pages 98–105, Piscataway, NJ, July 1999. IEEE Service Center.
3. J. D. Knowles and D. W. Corne. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Evolutionary Computation*, 8(2):149–172, 2000.
4. J. D. Knowles and D. W. Corne. M-PAES: A Memetic Algorithm for Multiobjective Optimization. In *Proceedings of the 2000 Congress on Evolutionary Computation (CEC 2000)*, Piscataway, NJ, 2000. IEEE. (To appear).
5. M. Laumanns, G. Rudolph, and H.-P. Schwefel. Approximating the Pareto Set: Concepts, Diversity Issues, and Performance Assessment. Technical Report CI-72/99, University of Dortmund, March 1999.
6. W. Mendenhall and R. J. Beaver. *Introduction to Probability and Statistics - 9th edition*. Duxbury Press, International Thomson Publishing, Pacific Grove, CA, 1994.
7. M. J. Oates and D. W. Corne. Investigating Evolutionary Approaches to Adaptive Database Management Against Various Quality of Service Metrics. In T. Bäck, M. Schoenauer, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature V*, pages 775–784. Springer, 1998.
8. M. Pilegaard Hansen and A. Jaszkiewicz. Evaluating the quality of approximations to the non-dominated set. Technical Report IMM-REP-1998-7, Technical University of Denmark, March 1998.
9. D. A. V. Veldhuizen and G. B. Lamont. Multiobjective Evolutionary Algorithm Test Suites. In J. Carroll, H. Haddad, D. Oppenheim, B. Bryant, and G. B. Lamont, editors, *Proceedings of the 1999 ACM Symposium on Applied Computing*, pages 351–357, San Antonio, Texas, 1999. ACM.
10. E. Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, November 1999. (See pp. 44–45).
11. E. Zitzler, K. Deb, and L. Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. Technical Report 70, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich, Switzerland, February 1999.
12. E. Zitzler and L. Thiele. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, November 1999.