2009 Ninth International Conference on Intelligent Systems Design and Applications

A Multiple Objective Particle Swarm Optimization Approach using Crowding Distance and Roulette Wheel

R. A. Santana, M. R. Pontes, C. J. A. Bastos-Filho
Department of Computing Systems, UPE
Recife - PE – Brazil 50720-001
ras@dsc.upe.br, mrp@dsc.upe.br, cjabf@dsc.upe.br

Abstract

This paper presents a multiobjective optimization algorithm based on Particle Swarm Optimization (MOPSO-CDR) that uses a diversity mechanism called crowding distance to select the social leaders and the cognitive leader. We also use the same mechanism to delete solutions of the external archive. The performance of our proposal was evaluated in five well known benchmark functions using four metrics previously presented in the literature. Our proposal was compared to other four multi objective optimization algorithms based on Particle Swarm Optimization, called m-DNPSO, CSS-MOPSO, MOPSO and MOPSO-CDLS. The results showed that the proposed approach is competitive when compared to the other approaches and outperforms the other algorithms in many cases.

1 Introduction

Many optimization problems may have to deal with multiple objectives and in some cases these objectives are conflicting. In this case, the problem will not have an unique solution. Instead of it, there will be several optimum solutions regarding on the compromise between the different objectives. The use of evolutionary algorithms and swarm intelligence algorithms have increased in recent years resulting in a wide variety of algorithms.

Particle Swarm Optimization (PSO) is a class of algorithms designed to solve optimization and search problems. PSO was proposed by Kennedy and Eberhart in 1995 [5], inspired by the social behavior of flocks of birds. In general, PSO is used to tackle with problems with one objective in hyper dimensional spaces where the variables are continuous. Despite the PSO was conceived to solve mono objective problems, due to its simplicity, some papers have proposed to extend the PSO to be applied to multi-objective problems.

This papers aims to improve a recently proposed approach by including a diversity mechanism called crowding distance to select the social and cognitive leaders and to delete solutions of the external archive.

This paper is organized as follow. In section 2 we present the basic concepts of Particle Swarm Optimization and Multi-objective optimization. In section 3 we review some Multi-objective Particle Swarm Optimization approaches. In section 4 we introduce our contribution. In sections 5 and 6 we present the simulation setup and the obtained results, respectively. In section 7 we give our conclusions.

2 Basic Concepts

In this section, we briefly review some basic concepts related to Particle Swarm Optimization and Multi objective Optimization. These concepts are crucial to understand our proposal.

2.1 Particle Swarm Optimization

Particle Swarm Optimization is a swarm intelligence population based algorithm [5]. The population is called swarm and the individuals are called particles. Each particle moves in the search space with an adaptable velocity looking for promising regions. Each particle i represents a possible solution to a problem and has three main attributes: the position in the search space $\vec{x_i}(t)$, the current velocity $\vec{v_i}(t)$ and the best position found by the particle during the search process $\vec{p_i}(t)$ so far. The particles move through the search space performing movements based on the best solution already found individually, and the best solution found by the neighborhood which they belong $\vec{p_{ig}}(t)$.

During each algorithm iteration, the particle´s velocities and positions are updated according to the equations (1) and (2), respectively. There are, at least, three different equation to update the velocity. The one used in this paper was developed by Shi and Eberhart [7] and involves the inertia factor w presented in the equation (1).

978-0-7695-3872-3/09 $26.00 © 2009 IEEE
DOI 10.1109/ISDA.2009.73

$$\vec{v_i}(t+1) = w\vec{v_i}(t) + c_1 r_1(\vec{p_i} - \vec{x_i}(t)) + c_2 r_2(\vec{p}_{ig}(t) - \vec{x_i}(t)), \quad (1)$$

$$\vec{x}_i(t+1) = \vec{x}_i + \vec{v}_i(t+1), \quad (2)$$

where $i = 1, ..., N$; c_1 and c_2 are two parameters called the cognitive and the social parameter that determine the relative influence of the social and cognition components; r_1 and r_2 are two random numbers generated by an uniform distribution in the interval [0,1].

2.2 Concepts of Multi Objective Optimization

A general multi-objective optimization minimization problems can be defined as:

$$\text{minimize } \vec{f}(\vec{x}) := [f_1(\vec{x}), f_2(\vec{x}), ..., f_k(\vec{x})] \quad (3)$$

subject to:

$$g_i(\vec{x}) \leq 0 \quad i = 1, 2, ..., m, \quad (4)$$

$$h_j(\vec{x}) = 0 \quad j = 1, 2, ..., p, \quad (5)$$

where $\vec{x} = (x_1, x_2, ..., x_n) \in \mathbb{R}^n$ is the vector on the decision search space; and $g_i(\vec{x})$ and $h_j(\vec{x})$ are the constraint functions of the problem.

Given two vectors $\vec{x}$, $\vec{y} \in \mathbb{R}^n$, $\vec{x}$ dominates $\vec{y}$ (denoted by $\vec{x} \prec \vec{y}$) if $\vec{x}$ is better than $\vec{y}$ in at least one objective and $\vec{x}$ is not worse than $\vec{y}$ in any objective. $\vec{x}$ is not dominated if does not exist another current solution $\vec{x}_i$ in the current population, such that $\vec{x}_i \prec \vec{x}$. The set of nondominated solutions in the objective space is known as Pareto Front ($\mathcal{PF}^*$).

3 Related Work

Some approaches have been proposed to extend the PSO for multiple objective problems. In this section we present some previous approaches that we used to compare with our proposal.

3.1 m-DNPSO

The m-DNPSO was proposed by Hu and Eberhart *et. al* [4]. In this algorithm, only one objective is optimized at each time step by using a scheme similar to lexographic ordering [2].

The multiple objectives are divided into two groups: f_1 and f_2. f_1 is defined as the neighborhood objective, f_2 is defined as the optimization objective. The selection of f_1 and f_2 is arbitrary.

To select the social leader, the algorithm has to evaluate the distance of the current particle to the other particles considering f_1. Using this information, the nearest m particles are selected (m is neighborhood size). Finally, the social leader is defined by the best solution in terms of f_2 values among the m neighbors.

The cognitive leader is updated only when a new solution dominates its current solution. An external archive (called "extend memory") is used to store the nondominated solutions.

This approach has two main drawbacks: Lexographic ordering tends to be useful only for two objectives and the performance may be sensitive to the order of the objectives.

3.2 MOPSO

This algorithm was proposed by Coello Coello *et. al* [3]. It is based on the idea of having an external archive in which every particle will deposit its experiences after each iteration. The external archive is updated considering a geographically-based system defined in terms of the objective function values for each particle. In this approach, the objective space explored is divided in hypercubes. Each hypercube receives a fitness value that depends on the number of particles inside the hypercube. A roulette-wheel selection is used to select a leader for each particle of the swarm. Once a hypercube is selected, one of the particles inside the hypercube is randomly chosen to be the leader. This approach also uses a mutation operator on the particles positions.

3.3 CSS-MOPSO

This algorithm was proposed by Chiu *et. al* [1]. In this algorithm, although there is no cognitive leader, there are two social leaders.

The g_{Best1} selection is performed based on the angle θ between the datum line, which connects the archive member and datum point c, and the line that connects the particle and the archive member that will be figured out. The archive member that presents the smallest angle with the i^{th} particle will be assigned as their local guide g_{Best1}.

The g_{Best2} selection is done according to the fitness value of a randomly selected objective f_i in each iteration. All the particles are sorted by their f_i fitness value. The CSS will assign the g_{Best2} using the following rule. For particles with even indexs, the archive member whose f_i fitness value is higher and nearest will be assigned as g_{Best2}. For particles with odd indexs, the archive member whose f_i fitness value is lower and nearest will be assigned as g_{Best2}.

3.4 MOPSO-CDLS

This algorithm was proposed by Tsou *et. al* [8] and is based on the approach proposed by Raquel *et. al* [6].

In this proposal, the crowding distance (CD) is used as a mechanism to select the leaders from the external archive. There are two possible situations: The social leader of a particle is randomly chosen among the 10% less crowded solutions, if the particle is dominated by these solutions; the leader is randomly chosen from the entire external archive, otherwise.

The cognitive leader of each particle is updated if the new position dominates the current cognitive leader. If these solutions are incomparable, the cognitive leader is randomly selected between these two options. This approach uses a local search mechanism in the external archive to improve the exploration abilities and to speed up the convergence.

4 Multiple Objective Particle Swarm Optimization with Crowding Distance and Roulette Wheel (MOPSO-CDR)

Our proposed algorithm is based on the approach proposed by Tsou *et. al* [8]. However, It incorporates the crowding distance mechanism and roulette wheel to select the social leader ($gBest$) and to prevent an excessive number of non dominated solutions in the external archive. Furthermore, MOPSO-CDR presents a novel procedure to update the cognitive leader ($pBest$). The pseudo code of our approach is presented in algorithm 1. The following subsections show the main features of our proposal.

Algorithm 1 Pseudo code for MOPSO-CDR

```
Initialize swarm
Initialize leaders in an external archive
Qualify leaders by using crowding distance
while stop criteria is not met do
  for each particle do
    Apply turbulence [3]
    Select leader (using crowding distance and roulette)
    Update velocity and position
    Evaluate fitness
    Update pBest (binary tournament)
  end for
  Update leaders in the external archive
  Qualify leaders by crowding distance
end while
Report results (External archive)
```

4.1 Social Leader Selection

For multiple objectives problems, a proper choice of the social leader ($gBest$) is crucial. It affects the convergence capability and the distribution of non dominated solutions along the Pareto front. The candidates for social leader are in the external archive. Each particle is enabled to seek for a different guide.

In MOPSO-CDR, the external archive is sorted by crowding distance before each iteration. For each particle, a social leader is selected by applying a roulette wheel in the external archive. Solutions in less crowded regions have more chance to be selected.

4.2 Cognitive Leader Selection

The cognitive leader ($pBest$) replacement rule is also important to the convergence and efectiveness of the algorithm. In the MOPSO-CDR, we developed a novel strategy to determine $pBest$. The cognitive leader of each particle is updated if the new position of the particle dominates $pBest$. If the new position and the $pBest$ are incomparable, the choice is made using the external archive. The algorithm search in the external archive for the solutions with minimum Euclidean distance for the $pBest$ and for the new position. If the closer solution to the new position in the external archive is in a less crowded region than the closer solution to the $pBest$ in the external archive, the new position will be chosen as the new $pBest$. Otherwise, the old $pBest$ remains.

4.3 Turbulence

PSO is known to have a good convergence speed. However, it may be harmful for multi objective optimization, because a PSO-based algorithm may converge to a false Pareto front (local optimum). Therefore, a mutation operator can help to avoid this problem by increasing the exploration ability of the particles.

The mutation operator used in our approach is the same used in the MOPSO [3] and it is applied at each iteration with a bounded influence. In the beginning, all the particles in the population are affected by the mutation operator. As the number of iterations increases, the percentage of affected particles decreases.

4.4 External archive

The main objective of the external archive is to record nondominated solutions found along the search process. The algorithm has to decide whether a certain solution should be added to the archive or not. The controller process used in MOPSO-CDR works as follows.

Initially, nondominated solutions are added to the external archive. After each iteration, nondominated solutions of the swarm are compared with the solutions in the external archive. The candidate solutions that are not dominated by the solutions of the external archive are added to it. Then, the dominated solutions are removed from the external archive.

The external archive has a maximum number of solutions. If this number is exceeded after the end of each iteration, solutions in more crowded regions are removed from the repository using the crowding distance criterion.

5 Simulation Setup

5.1 Benchmark functions

Zitzler *et al*. [10] proposed a set of benchmark test problems which can be used to compare multi-objective optimization approaches. We used the following test functions of this benchmark ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6. We did not used the ZDT5 function since it is designed to deal with string of binary numbers and PSO is not suitable, in principle, for this type of application.

5.2 Metrics

There are some metrics that can be used to quantify the quality of the Pareto Fronts obtained by the algorithms. The following metrics will be used in this paper: Hypervolume, Spacing, Maximum spread and Coverage. One should attempt that each metric highlight a different aspect of the Pareto Front.

5.2.1 Hypervolume

The hypervolume (HV) was proposed by Zitzler and Thiele [9].

In order to explain this concept, consider an optimization problem with two objectives. The hypervolume is defined by the area in the objective search space covered by the obtained Pareto front ($\mathcal{P}_a^*$) (i.e., the "area under the curve").

Consider a rectangle bounded by one point $(f_1(\vec{x}); f_2(\vec{x}))$ that belongs to the Pareto front and the origin. Suppose that each point in the Pareto front generates a rectangle in the objective space. The hypervolume corresponds to the area formed by the union of all those rectangles.

It is also possible to generalize this concept to problems with n-objectives, by using the following equation:

$$HV = \left\{ \bigcup_i a_i \,|\, x_i \in \mathcal{P}_a^* \right\}, \tag{6}$$

where x_i is a non dominated vector in $\mathcal{P}_a^*$ and a_i is the Hypervolume determined by the components of x_i and the origin.

5.2.2 Spacing

The goal is to measure the spread (distribution) of non dominated solutions throughout the Pareto front. Actually, it measures the variance of the distance between adjacent nondominated solutions and can be evaluated by equation (7).

$$S = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(\bar{d} - d_i)^2}, \tag{7}$$

where $d_i = min_j(|f_1^i(\vec{x}) - f_1^j(\vec{x})| + |f_2^i(\vec{x}) - f_2^j(\vec{x})|)$, $i, j = 1, ..., n$, $\bar{d}$ is the mean distance between all the adjacent solutions and n is the number of non dominated solutions in the Pareto front.

A value equal to zero means that all the solutions are equidistantly spaced in the Pareto front.

5.2.3 Maximum Spread

This metric was proposed by Zitzler *et. al* [10] and evaluates the maximum extension covered by the nondominated solutions in the Pareto front. In a two objectives problem, the Maximum Spread corresponds to the Euclidean distance between the two farther solutions.

$$MS = \sqrt{\sum_{m=1}^{M}(max_{i=1}^{n} f_m^i - min_{i=1}^{n} f_m^i)^2}, \tag{8}$$

where n is number of solutions in the Pareto front and M is the number of objectives in a given problem. One should note that higher values indicate better performance.

5.2.4 Coverage

The two set coverage metric (C) proposed by Zitzler *et. al* [11] [12] [9] maps the ordered pair (A, B) to the interval [0,1] using the following equation:

$$C(A,B) = \frac{|\{b \in B; \exists a \in A : a \succeq b\}|}{|B|} \tag{9}$$

The value $C(A,B) = 1$ means that all solutions in B are weakly dominated by A. On the other hand, $C(A,B) = 0$ means that none of the solutions in B are weakly dominated by A.

One should note that both $C(A,B)$ and $C(B,A)$ have to be evaluated, since $C(A,B)$ is not necessarily equal to $1 - C(B,A)$.

If $0 < C(A,B) < 1$ and $0 < C(B,A) < 1$, then neither A weakly dominates B nor B weakly dominates A. Thus, the sets A and B are incomparable, it means that A is not worse than B and vice-versa.

5.3 Parameter Settings

Each algorithm has its own set of parameters.

In the MOPSO, the mutation rate is 0.5, the number of divisions for the adaptive grid is 30 and inertia factor linearly decreases from 0.4 to 0.0 [3].

In the MOPSO-CDLS, sigma is equal to 0.2 in the local search operator and inertia factor linearly decreases from 0.9 to 0.4 [8].

In the m-DNPSO, the most simple objective function was assigned to f_1 and the most complex to f_2. $m = 10$ and the inertia factor is randomly generated for each iteration in the interval [0.5,1.0] [4].

In the CSS-MOPSO, the standard deviation for the Gaussian mutation is 0.01 and inertia factor linearly decreases from 0.9 to 0.4 [1].

MOPSO-CDR used a mutation rate of 0.5. Since our proposal is based on MOPSO-CDLS, one could expect the same inertia factor scheme. However, we observed that the scheme linearly decreasing the inertia factor from 0.4 to 0.0 performed better.

In all the cases, we used 20 particles, a maximum of 200 solutions in the external archive. 200,000 fitness evaluation were executed in each trial. We used the cognitive and social acceleration constants equal to 1.49445, when it is applied. Each situation was evaluated 30 times. The results are presented in terms of mean value and standard deviation.

6 Results

Tables I to V show the simulation results for all the algorithms considering the four performance metrics.

From tables 1 and 2, one can note that MOPSO-CDR and CSS-MOPSO outperformed m-DNPSO, MOPSO and MOPSO-CDLS in terms of Hypervolume, Spacing and Coverage for ZDT1 and ZDT2 problems, respectively. m-DNPSO achieved slightly better Maximum Spreading. Comparing the coverage of CSS-MOPSO and MOPSO-CDR, one can see that most part of nondominated solutions of MOPSO-CDR dominates the nondominated solutions of CSS-MOPSO, but none of the MOPSO-CDR nondominated are dominated by CSS-MOPSO nondominated.

Table 3 presents the simulation results for ZDT3 problem. One can note that MOPSO-CDR outperformed the other algorithms in terms of Hypervolume, Spacing and Coverage. m-DNPSO achieved slightly better Maximum Spreading.

Table 1. Mean value and standard deviation - ZDT 1 problem.

Alg.	Hyper Volume	Spacing	Max. Spread	Cover. CDR,*	Cover. *,CDR
MOPSO	0.36 (0.002)	0.0046 (5E-4)	1.425 (0.005)	1.0 (0.0)	0.0 (0.0)
m-DNPSO	0.713 (0.053)	0.0457 (0.014)	1.54 (0.065)	1.0 (0.0)	0.0 (0.0)
MOPSO CDLS	0.39 (0.003)	0.0042 (6E-4)	1.44 (0.005)	1.0 (0.0)	0.0 (0.0)
CSS MOPSO	0.34 (0.002)	0.0023 (1E-4)	1.42 (0.002)	0.99 (0.003)	0.0 (0.0)
MOPSO CDR	0.33 (3E-5)	0.0033 (2E-4)	1.41 (0.0)	- -	- -

Table 2. Mean value and standard deviation - ZDT 2 problem.

Alg.	Hyper Volume	Spacing	Max. Spread	Cover. CDR,*	Cover. *,CDR
MOPSO	0.69 (0.001)	0.006 (0.001)	1.396 (0.015)	1.0 (0.0)	0.0 (0.0)
m-DNPSO	0.94 (0.06)	0.054 (0.017)	1.29 (0.037)	1.0 (0.0)	0.0 (0.0)
MOPSO CDLS	0.716 (0.0035)	0.006 (0.001)	1.39 (0.004)	1.0 (0.0)	0.0 (0.0)
CSS MOPSO	0.674 (0.001)	0.0035 (7E-4)	1.41 (8E-4)	0.978 (0.021)	0.0 (0.0)
MOPSO CDR	0.66 (3E-5)	0.0032 (1E-4)	1.41 (0.0)	- -	- -

Table 3. Mean value and standard deviation - ZDT 3 problem.

Alg.	Hyper Volume	Spacing	Max. Spread	Cover. CDR,*	Cover. *,CDR
MOPSO	0.950 (0.004)	0.005 (4E-4)	1.976 (0.008)	1.0 (0.0)	0.0 (0.0)
m-DNPSO	1.296 ((0.088)	0.045 (0.016)	2.068 (0.146)	1.0 (0.0)	0.0 (0.0)
MOPSO CDLS	1.006 (0.009)	0.006 (9E-4)	1.988 (0.015)	1.0 (0.0)	0.0 (0.0)
CSS MOPSO	0.953 (0.008)	0.003 (7E-4)	1.983 (0.006)	0.999 (8E-4)	0.0 (0.0)
MOPSO CDR	0.920 (1E-4)	0.003 (2E-4)	1.967 (2E-5)	- -	- -

Table 4 presents the simulation results for ZDT4 problem. MOPSO-CDR outperformed the other algorithms in terms of Hypervolume and Spacing. CSS-MOPSO

achieved a better Maximum Spreading. From the Coverage results, one can conclude that MOPSO-CDR and MOPSO achieved the best sets of nondominated solutions. However, the MOPSO-CDR nondominated solutions are slightly better.

Table 4. Mean value and standard deviation - ZDT 4 problem.

Alg.	Hyper Volume	Spacing	Max. Spread	Cover. CDR,*	Cover. *,CDR
MOPSO	0.631 (0.526)	0.006 (0.0014)	1.54 (0.18)	0.51 (0.429)	0.4 (0.49)
m-DNPSO	2.157 (0.935)	0.04 (0.037)	1.94 (0.29)	0.97 (0.18)	0.03 (0.18)
MOPSO CDLS	4.82 (2.174)	0.005 (9E-4)	2.70 (0.46)	1.0 (0.0)	0.0 (0.0)
CSS MOPSO	5.38 (2.54)	0.005 (0.0012)	2.80 (0.525)	1.0 (0.0)	0.0 (0.0)
MOPSO CDR	0.57 (0.26)	0.003 (3E-4)	1.52 (0.109)	- -	- -

Table 5 presents the simulation results for ZDT6 problem. MOPSO-CDR outperformed the other algorithms in terms of Spacing. CSS-MOPSO achieved a better Maximum Spreading. MOPSO and m-DNPSO achieved the best hypervolumes. From the Coverage results, one can conclude that MOPSO-CDR results are better when compared to the other algorithms.

Table 5. Mean value and standard deviation - ZDT 6 problem.

Alg.	Hyper Volume	Spacing	Max. Spread	Cover. CDR,*	Cover. *,CDR
MOPSO	1.261 (0.386)	0.129 (0.122)	3.180 (1.400)	0.432 (0.294)	0.072 (0.073)
m-DNPSO	1.279 (0.506)	0.126 (0.108)	3.203 (1.732)	0.731 (0.153)	0.065 (0.064)
MOPSO CDLS	1.717 (0.519)	0.186 (0.145)	4.632 (1.816)	0.741 (0.150)	0.102 (0.086)
CSS MOPSO	2.051 (0.697)	0.234 (0.153)	5.571 (2.046)	0.121 (0.053)	0.013 (0.049)
MOPSO CDR	1.670 (0.300)	0.088 (0.056)	4.636 (1.053)	- -	- -

7 Conclusion

This paper presented a multi-objective PSO. The rules used to select the social leader and cognitive leader are based on the crowding distance and roulette wheel. We developed a novel strategy to determine the cognitive leader based on a comparison using the external archive. According to the results, our proposal can generate better nondominated solutions than the other algorithms. One advantage of our proposal is the use of few parameters. We believe that the MOPSO-CDR may perform even better than the other algorithms in problems with more than two objectives.

Acknowledgment

The authors thank the Polytechnic School of Pernambuco, FACEPE and CNPq funding agencies.

References

[1] S.-Y. Chiu, T.-Y. Sun, S.-T. Hsieh, and C.-W. Lin. Cross-searching strategy for multi-objective particle swarm optimization. In *IEEE Congress on Evolutionary Computation*, pages 3135–3141. IEEE.

[2] Coello, G. B. Lamont, and D. A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Genetic and Evolutionary Computation. Springer, New York, 2nd edition, 2007.

[3] C. Coello, G. Pulido, and M. Lechuga. Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 8(3):256–279, 2004.

[4] X. Hu, R. Eberhart, and Y. Shi. Particle swarm with extended memory for multiobjective optimization. In *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE*, pages 193–197, 2003.

[5] J. Kennedy and R. Eberhart. Particle swarm optimization. volume 4, pages 1942–1948 vol.4, 1995.

[6] C. R. Raquel and P. C. Naval, Jr. An effective use of crowding distance in multiobjective particle swarm optimization. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 257–264, New York, NY, USA, 2005. ACM.

[7] Y. Shi and R. C. Eberhart. Parameter selection in particle swarm optimization. In *EP '98: Proceedings of the 7th International Conference on Evolutionary Programming VII*, pages 591–600, London, UK. Springer-Verlag.

[8] C. Tsou, S. Chang, and P. Lai. Using Crowding Distance to Improve Multi-Objective PSO with Local Search.

[9] E. Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, ETH Zurich, Switzerland, 1999.

[10] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary computation*, 8(2):173–195, 2000.

[11] E. Zitzler and L. Thiele. Multiobjective Optimization Using Evolutionary Algorithms - A Comparative Case Study. In *Conference on Parallel Problem Solving from Nature (PPSN V)*, pages 292–301, Amsterdam, 1998.

[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, 1999.