© International Journal of Combinatorial Optimization Problems and Informatics, Vol. 8, No. 1, Jan-April 2017, pp. 12-18. ISSN: 2007-1558.

A Neighborhood Operator for Continuous Multi-Objective Optimization Problems

Alejandro Santiago[1,2], José Carlos Soto Monterrubio[1,2], J. David Terán-Villanueva[1], Héctor Joaquín Fraire Huacuja[1], Juan Javier González Barbosa[1], Claudia Gómez Santillán[1]
[1]Instituto Tecnológico Nacional de México, Instituto Tecnológico de Ciudad Madero
[2]Universidad de Cádiz
E-mail: alejandro.santiagopi@mail.uca.es, soto190@gmail.com, automatas2002@yahoo.com.mx, jjgonzalezbarbosa@hotmail.com, cggs71@hotmail.com.

A la memoria de Jesús Vicente Flores Morfín

Abstract. Path-metaheuristics have been used successfully in combinatorial optimization. However, in continuous optimization problems, the lack of neighborhood definitions makes them difficult to design and implement. This paper proposes a neighborhood operator based on first order linear approximation of the gradient. In order to adapt the linear approximation to multi-objective optimization, we use the multi-objective decomposition approach so the operator can be used for single and multi-objective continuous optimization problems. The proposed approach is validated using a Threshold Accepting algorithm based on decomposition and a set of benchmark problems for multi-objective optimization. Results show a significant improvement over Pareto lineal sets.

1. Introduction

Multi-objective optimization (MOO) based on decomposition [1] is an approach that researchers use to avoid the problems of Pareto dominance, some of the advantages are less complex algorithms and no need of dominance verification of the solutions found. In MOO all solutions that are non-dominated are equally good. MOO based on decomposition guides the search using aggregative objective functions as fitness function; the main idea of MOO based on decomposition is transforming the Multi-objective problem into many single objective problems to achieve different Pareto optimal solutions. This is done using weighted aggregative objective functions. The current research on decomposition methods has been focused on population searches as evolutionary algorithms (MOEA/D [2], MOEA/D-DE [3]), memetic algorithms (MOGLS [4], MOEA/D-SQA [5]) and swarm intelligence (MOEA/D-ACO [6], dMOPSO [7], D2MOPSO [8]) among others. Early attempts to study path-search based on decomposition were presented in [9, 10]. In [9] Alhindi et al. presents a Tabu Search for multi-objective permutation flow shop scheduling problems while in [10] Li et al. uses the multi-objective versions of knapsack and traveling salesman. In combinatorial optimization, there are well-known operators [11]; i.e., the swap and insertion operators are very effective for permutation based problems like the linear ordering problem [12, 13] while the boundary or sequential changes are effective for combinatorial problems [14]. The main issue with continuous optimization problems is the concept of movement among neighbor solutions, which is not clear. In this paper we will use the Threshold Accepting (TA) algorithm which is a variant of the Simulated Annealing algorithm, and we will name it Threshold Accepting based on Decomposition algorithm (TAD). We selected this algorithm in order to use a simple path-search metaheuristic and prove the efficiency of the proposed neighborhood. The main contribution of our work is a new neighborhood operator based on linear approximations of the gradient. The remaining of this paper is organized as follows. Section 2 shows the definitions of multi-objective optimization and aggregative objective functions. Section 3 describes the three different linear approximations of the gradient. Section 4 describes the proposed approach and an algorithm to prove the neighborhood performance. The experimentation setup is shown in Section 5. A description of the experimental results can be found on Section 6. Section 7 contains the conclusions and future work.

Received Jun 19, 2016 / Accepted Dec 10, 2016

2. Background concepts

This section presents basic concepts of multi-objective optimization and aggregative objective functions from [1, 22].

Definition 1. *A Multi-objective optimization problem (*MOP*).*
Given a vector function $\vec{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), \ldots\ldots\ldots, f_k(\vec{x})]$ and its feasible solution space Ω, the MOP consists in finding a vector $\vec{x} \in \Omega$ that optimizes the vector function $\vec{f}(\vec{x})$. Without loss of generality we will assume only minimization functions.

Definition 2. *Pareto dominance.*
A vector $\vec{x}$ dominates $\vec{x}'$ (denoted by $\vec{x} \prec \vec{x}'$) if $f_i(\vec{x}) \leq f_i(\vec{x}')$ for all i functions in $\vec{f}$ and there is at least one i such that $f_i(\vec{x}) < f_i(\vec{x}')$.

Definition 3. *Pareto optimal.*
A vector $\vec{x}^*$ is Pareto optimal if not exist $\vec{x'} \in \Omega$ *such that* $\vec{x'} \prec \vec{x^*}$.

Definition 4. *Pareto optimal set.*
Given a MOP, the Pareto optimal set is defined as $P^* = \{\vec{x}^* \in \Omega\}$.

Definition 5. *Pareto front.*
Given a MOP and its Pareto optimal set P^*, the Pareto front is defined as $PF^* = \{\vec{f}(\vec{x}) \mid \vec{x} \in P^*\}$.

Definition 6. *Aggregative objective function (*AOF*).*
Given a vector function $\vec{f}(\vec{x})$, an AOF is a function that maps $\vec{f}(\vec{x}) \rightarrow \mathbb{R}$.

Definition 7. *Weighted AOF.*
Given a vector of weights $\vec{w} = (w_1, w_2, w_3, \ldots\ldots\ldots, w_k)$ and a vector function $\vec{f}(\vec{x})$, a weigthed AOF γ is a function that $\gamma(\vec{f}(\vec{x}), \vec{w}) \rightarrow \mathbb{R}$.

3. Neighborhood operator based on linear approximations of the gradient

The gradient vector ∇f of a scalar valued function f (as the weighted AOFs) on point $\vec{x}$, indicates the direction where the scalar field f changes most quickly. There exist three different ways of compute the linear approximations of the gradient: forward (Equation 1 and 3), backward (Equation 2 and 4) and center (Equation 1, 2 and 5). To compute the approximations a perturbation value Δ is necessary. In [15] Hughes proposed the use of forward approximation in order to guide the search. The problem using only forward gradient is that the decision variables can only be updated by positive increments as in Equation 1, leading the search to stagnation. Also the selection of Δ is not trivial, lower values of Δ will produce lesser exploration of the search space, while higher values will create unfeasible solutions.

$$x_i' \leftarrow x_i + \Delta \quad (1)$$

$$x_i'' \leftarrow x_i - \Delta \quad (2)$$

$$\nabla f_i(\vec{x}) \approx \frac{f(\vec{x}) - f(\vec{x}\prime)}{\Delta} \quad (3)$$

$$\nabla f_i(\vec{x}) \approx \frac{f(\vec{x}\prime\prime) - f(\vec{x})}{\Delta} \quad (4)$$

$$\nabla f_i(\vec{x}) \approx \frac{f(\vec{x}\prime) - f(\vec{x}\prime\prime)}{2\Delta} \quad (5)$$

The computation of the approximation of the gradient ∇f increases with the number of decision variables plus the computational cost of founding a feasible value of Δ for n variables both operations are of $O(n)$ complexity. In order to explore the search space and reduce the computational times only one decision variable from the original vector is modified as suggested in [15]. Algorithm 1 shows the entire neighborhood application procedure given a solution $\vec{x}$ and their variables bounds, the procedure

return a neighbor solution $\vec{x}'$, the central calculation use the temporal solutions $\vec{z}$ and $\vec{y}$. The repair method is very simple, if the decision variable is greater than the upper bound it is set to the upper bound, if the decision variable is smaller than the lower bound is set to the lower bound.

Require: a solution $\vec{x}$
  Select a random $x_i \in \vec{x}$
  Calculate Δ for x_i
  if $x_i = UpperBound \quad || \quad x_i = LowerBound$ **then**
    $\Delta \leftarrow UpperBound$
  else
    $\Delta \leftarrow min(UpperBound - x_i, x_i - LowerBound)$
  end if
  switch x_i **do**
    case $UpperBound$
      $x'_j \leftarrow x_j \quad \forall j \neq i$
      $x'_i \leftarrow LowerBound$
      $x'_i \leftarrow \frac{f(\vec{x}) - f(\vec{x}')}{\Delta}$
    case $LowerBound$
      $x'_j \leftarrow x_j \quad \forall j \neq i$
      $x'_i \leftarrow UpperBound$
      $x'_i \leftarrow \frac{f(\vec{x}) - f(\vec{x}')}{\Delta}$
    case $x_i \neq UpperBound \wedge x_i \neq LowerBound$
      $\Delta \leftarrow min(UpperBound - x_i, x_i - LowerBound)$
      $x'_j \leftarrow x_j \quad \forall j \neq i$
      $z_j \leftarrow x_j \quad \forall j \neq i$
      $y_j \leftarrow x_j \quad \forall j \neq i$
      $z'_i \leftarrow x_i + \Delta$
      $y'_i \leftarrow x_i - \Delta$
      $x'_i \leftarrow \frac{f(\vec{z}) - f(\vec{y})}{\Delta}$
  $Repair(\vec{x}')$ ▷ with the upper or lower bound of the decision variable.
  return $\vec{x}'$

Algorithm 1. Function *LinearApproximationGradient*()

4. Threshold accepting based on decomposition (TAD)

The Multi-objective TAD is a hybrid between the decomposition and Threshold Accepting framework. A set of vector weights W is required by the AOF to guide the search [1]. TAD also benefits from the mutated solutions $\vec{x}'$ used to compute the neighbor $\vec{x}''$ to update the current problem cluster c_i or the entire population as in MOEA/D-DE [3]. The clusters c_i can be overlapped, as the Euclidean distance is used and every cluster is formed with the T closest $\vec{w}_j$ to $\vec{w}_i$ [3]. The weighted AOF used in this work is the weighted Tchebycheff, the same weighted AOF that MOEA/D-DE implements. The following algorithm describes TAD based on decomposition with a neighborhood operator based on linear approximations of the gradient.

1: Compute distances between any two $w_i \in W$
2: **for** $\forall \vec{w}_i \in W$ **do**
3: $C_i \leftarrow$ the T nearest problems to w_i
4: $\vec{x}_i \leftarrow$ an initial solution for the problem $\vec{w}_i$
5: **end for**
6: $Q \leftarrow$ a high value
7: $Q_{min} \leftarrow$ a low value
8: $r \leftarrow$ reduction factor .95 for 5%
9: **while** $Q > Q_{min}$ **do**
10: **for** $i = 1$ to $|W|$ **do**
11: **for** $j = 1$ to $maxsteps$ **do**
12: $\vec{x}' \leftarrow PolynomialMutation(\vec{x}_i)$
13: $\vec{x}'' \leftarrow LinearApproximationGradient(\vec{x}')$
14: **if** $\gamma(\vec{f}(\vec{x}''), \vec{w}_i)) - \gamma(\vec{f}(\vec{x}_i), \vec{w}_i)) < Q$ **then**
15: $\vec{x}_i \leftarrow \vec{x}''$
16: **end if**
17: $UpdateClusters(\vec{x}')$;
18: $UpdateClusters(\vec{x}'')$;
19: $maxsteps++$
20: **end for**
21: **end for**
22: $Q \leftarrow Q \cdot r$
23: **end while**

Algorithm 2. Threshold accepting algorithm based on decomposition (TAD)

As shown in Algorithm 2 the main while loop (lines 9-23) controls the stop criteria, reached when the threshold Q is equal or less than the minimum threshold value Q_{min}. Every vector weight $\vec{w}_i \in W$ has associated a best current solution $\vec{x}_i$, which is visited in the inner loop (lines 10-21) and is mutated *maxsteps* times in two steps (lines 11-20), the first step performs polynomial mutation [16] to add diversity and avoid the exclusive use of linear movements in a temporal solution $\vec{x}'$, the second step computes their neighbor $\vec{x}''$using the proposed operator. If $\vec{x}''$ objective value is better than the threshold $Q + \gamma(\bar{f}(\vec{x}_i), \vec{w}_i)$ then $\vec{x}_i$ is replaced with $\vec{x}''$ (lines 14-15). It is important to notice that when the gradient is computed backward or forward the temporal solution x' is only one, on the other hand when central gradient is computed two temporal solutions are used (forward and backward) as in Equation 5, every temporal solutions is used later in the $UpdateClusters$ method, as the update step in MOEA/D-DE [3].

5. Experimentation

For the experimentation, three algorithms from the state of the art were selected NSGA-II, GDE3, and MOEA/D-DE. The quality indicators computed are Generational Distance [17], Generalized Spread [18] and Spacing [19]. The TAD parameters are: the threshold start value $Q = 1$, threshold end value Q_{min}= 1E-10, the reduction factor *r*=0.95, the decomposition mating/update range σ=0.9, the number of neighbor solutions to visit *maxsteps*=10, which is equivalent to 449 generations in classical evolutionary algorithms. The experimentation was performed with the original configuration parameters proposed by their authors. One hundred independent runs were performed over each treated problem. The experimentation benchmarks are ZDT [20], LZ09 [3] and DTLZ [21]. The algorithm and problems implementations were taken from the framework jMetal 4.5 [23] and the experimentation was carried out on an IBM x3100 M4 computer with Xeon Quad Core 3.0 Ghz with 2 GB of RAM running FreeBSD 10 as operative system.

6. Results

The averages of the quality indicators computed are shown in Tables 1 to 3. Although the best performance is not always achieved by a decomposition method, this work analyzes the difference between the TAD which includes the proposed neighborhood operator and MOEA/D-DE. The problems of the DTLZ family were configured with the optimal parameters suggested in [21] with three objectives. The problems with Pareto linear shapes in DTLZ family are DTLZ1 and DTLZ2, for

both problems our proposed TAD gets the best uniformity (spacing indicator) among the solutions found. For the second family of problems ZDT, the same behavior occurs. Almost all the problems have linear Pareto shapes because the first objective f_1 is equal to the first variable x_1, except for ZDT5 which is a binary problem and ZDT6, were MOEA/D-DE outperforms our proposed TAD. The LZ09 family of problems is a set of hard optimization problems because of its complicated Pareto shapes. Our proposed TAD has a lower performance than MOEA/D-DE over these set or problems, LZ09_F8 presents a special interest to us because our proposed TAD outperforms MOEAD/-DE. The main reason of the hardness of the LZ09 problems when solved with TAD is the threshold value and reduction factor, LZ09 problems have very small objective values which needs a smoother reduction factor and a smaller final threshold value, in order to be sensitive to the changes in the objective functions values. Finally, the non-parametric statistical test of Kruskal-Wallis produces significant differences, above 95% of confidence for all the benchmark problems. The Wilcoxon rank sum test also shows a significant difference between MOEA/D-DE and TAD, above 95% for all the benchmark problems except for the GD of DTLZ6 where both algorithms were reported as equivalent. The tables in this section show in light gray the best decomposition method. The abbreviations are: Generational Distance GD, Generalized Spread GS and Spacing SPA.

Table 1. DTLZ results

	GDE3	MOEAD/DE	NSGAII	TAD	GDE3	MOEAD/DE	NSGAII	TAD	GDE3	MOEAD/DE	NSGAII	TAD
Problem	DTLZ1				DTLZ2				DTLZ3			
GD	0.00021	0.78845	4.95647	0.85970	0.00049	0.00355	0.00065	0.00340	1.88681	2.84279	23.71066	3.09526
GS	0.41590	0.44565	0.02977	0.36167	0.37782	0.41316	0.37726	0.41626	0.03114	0.38985	0.00563	0.32703
SPA	0.00905	16.31855	8.36979	14.65828	0.02212	0.09713	0.02440	0.08509	1.71453	75.18061	24.43615	79.32565
Problem	DTLZ4				DTLZ5				DTLZ6			
GD	0.00190	0.00440	0.00204	0.00578	0.00010	0.00252	0.00012	0.00227	0.00022	0.01694	0.24529	0.01690
GS	0.46959	0.74281	0.45169	0.66884	0.53090	0.76705	0.41455	0.77189	0.54077	0.78889	0.01380	0.75920
SPA	0.02220	0.04189	0.02442	0.05808	0.00086	0.07222	0.00209	0.06175	0.00085	0.48096	0.20036	0.46698
Problem	DTLZ7											
GD	0.00119	0.03701	0.00878	0.02180								
GS	0.51713	0.66791	0.48206	0.66518								
SPA	0.02717	0.21473	0.03303	0.27982								

Table 2. ZDT results

	GDE3	MOEAD/DE	NSGAII	TAD	GDE3	MOEAD/DE	NSGAII	TAD	GDE3	MOEAD/DE	NSGAII	TAD
Problem	ZDT1				ZDT2				ZDT3			
GD	0.00008	0.01456	0.00057	0.00870	0.00003	0.01481	0.00112	0.00765	0.00019	0.01458	0.00112	0.00765
GS	0.51786	0.59042	0.13623	0.62051	0.52063	0.53661	0.12460	0.60206	0.54016	0.82648	0.56377	0.73640
SPA	0.00094	0.25546	0.00240	0.15402	0.00093	0.25640	0.00407	0.13245	0.00103	0.25727	0.00250	0.17088
Problem	ZDT4				ZDT6							
GD	0.43038	0.61116	0.07702	0.26028	0.00025	0.00616	0.03728	0.01690				
GS	0.09561	0.57426	0.26941	0.18483	0.72209	0.75978	0.19898	0.91813				
SPA	0.18495	10.58875	0.05712	4.51059	0.00095	0.10550	0.02040	0.29277				

Table 3. LZ09 results

	GDE3	MOEAD/DE	NSGAII	TAD	GDE3	MOEAD/DE	NSGAII	TAD	GDE3	MOEAD/DE	NSGAII	TAD
Problem	LZ09_F1				LZ09_F3				LZ09_F4			
GD	0.00074	0.00356	0.00207	0.01236	0.00159	0.00040	0.00331	0.00419	0.00156	0.00343	0.00229	0.00372
GS	0.25596	0.54724	0.63339	0.36029	0.64526	0.91408	0.51263	1.04331	0.71059	1.07755	0.60250	1.18876
SPA	0.00381	0.06261	0.00662	0.21433	0.00880	0.00612	0.00817	0.06860	0.00872	0.02448	0.00608	0.06573
Problem	LZ09_F5				LZ09_F6				LZ09_F7			
GD	0.00112	0.00109	0.00219	0.00442	0.17813	0.03286	0.09175	0.01986	0.16661	0.03111	0.08978	0.02332
GS	0.69486	1.13480	0.53689	1.02732	0.25007	1.03308	0.42151	0.59000	0.16406	0.52020	0.26105	0.50653
SPA	0.00668	0.00648	0.00555	0.07285	0.21476	0.54351	0.15077	0.29373	0.20300	0.48539	0.10816	0.30646
Problem	LZ09_F8											
GD	0.01521	0.01576	0.02666	0.01483								
GS	0.39725	0.42393	0.36577	0.35093								
SPA	0.00271	0.00991	0.01731	0.09001								

7. Conclusions

We recommend the use of the neighborhood operator based on linear approximations of the gradient especially for linear Pareto shapes. The parameter Q_{min} has to be smaller enough in decimal precision to detect the changes in the objective functions of the treated problem. For problems with differences below 30 decimals (Q_{min}=1E-30) we recommend a smoother reduction factor as 999E-3. The inclusion of polynomial mutation allows the algorithm to work with not only linear Pareto shapes, as show in LZ09_F8, DTLZ5, DTLZ6. The weakness of the proposed approach is observed when very small changes in the objective values happen, then a smoother reduction factor and smaller final threshold is needed. However, those are natural deficiencies of the threshold approach, while our proposal remains independent from a particular path-search algorithm and was proved to yield good performance.

The impact of higher order linear approximations and how it affects the final Pareto front representation still need to be researched. Path-search based on decomposition for multi-objective optimization is a promising research area, with many traditional methods not which have not been researched yet. We invite researchers to test the proposed neighborhood operator as an alternative to their operators and encourage them to implement new operators to increase the knowledge on path-search algorithms for continuous optimization problems.

Acknowledgements

The authors would like to acknowledge with appreciation and gratitude to CONACYT, TECNM and PRODEP. A. Santiago, J. Carlos Soto Monterrubio J. and David Terán-Villanueva would like to thank CONACyT for the supports 360199, 414092 and 177007.

References

1. A. Santiago, H. Huacuja, B. Dorronsoro, J. Pecero, C. Santillan, J. Barbosa, and J. Monterrubio. A survey of decomposition methods for multi-objective optimization. In O. Castillo, P. Melin, W. Pedrycz, and J. Kacprzyk, editors, Recent Advances on Hybrid Approaches

for Designing Intelligent Systems, volume 547 of Studies in Computational Intelligence, pages 453–465. Springer International Publishing, 2014.
2. Q. Zhang and H. Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. Evolutionary Computation, IEEE Transactions on, 11(6):712–731, Dec 2007.
3. H. Li and Q. Zhang. Multiobjective optimization problems with complicated Pareto sets, moea/d and nsga-ii. Evolutionary Computation, IEEE Transactions on, 13(2):284–302, April 2009.
4. H. Ishibuchi and T. Murata. A multi-objective genetic local search algorithm and its application to flowshop scheduling. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, 28(3):392–403, 1998.
5. Y.-Y. Tan, Y.-C. Jiao, H. Li, and X.-K. Wang. Moea/d-sqa: a multi-objective memetic algorithm based on decomposition. Engineering Optimization, 44(9):1095–1115, 2012.
6. L. Ke, Q. Zhang, and R. Battiti. Moea/d-aco: A multiobjective evolutionary algorithm using decomposition and antcolony. Cybernetics, IEEE Transactions on, 43(6):1845–1859, Dec 2013.
7. S. Zapotecas Martínez and C. A. Coello Coello. A multi-objective particle swarm optimizer based on decomposition. In Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, GECCO '11, pages 69–76, New York, NY, USA, 2011. ACM.
8. N. Al Moubayed, A. Petrovski, and J. McCall. D2mopso: Multi-objective particle swarm optimizer based on decomposition and dominance. In Proceedings of the 12th European Conference on Evolutionary Computation in Combinatorial Optimization, EvoCOP'12, pages 75–86, Berlin, Heidelberg, 2012. Springer-Verlag.
9. A. Alhindi and Q. Zhang. Moea/d with tabu search for multiobjective permutation flow shop scheduling problems. In Evolutionary Computation (CEC), 2014 IEEE Congress on, pages 1155–1164, July 2014.
10. H. Li and D. Landa-Silva. An adaptive evolutionary multi-objective approach based on simulated annealing. Evol. Comput., 19(4):561–595, Dec. 2011.
11. Pierre Hansen, Nenad Mladenović, Variable neighborhood search: Principles and applications, European Journal of Operational Research, Volume 130, Issue 3, 1 May 2001, Pages 449-467, ISSN 0377-2217.
12. D. Teran et al. Hybrid grasp with composite local search and path-relinking for the linear ordering problem with cumulative costs. International Journal of Combinatorial Optimization Problems, journal 1, volume 3, pages 21–30, 2012.
13. T. Schiavinotto and T. Stützle. The linear ordering problem: Instances, search space analysis and algorithms. Journal of Mathematical Modelling and Algorithms, 3(4):367–402, 2005.
14. J. E. Pecero, H. J. F. Huacuja, P. Bouvry, A. A. S. Pineda, M. C. L. Locés and J. J. G. Barbosa, "On the energy optimization for precedence constrained applications using local search algorithms," High Performance Computing and Simulation (HPCS), 2012 International Conference on, Madrid, 2012, pp. 133-139. doi: 10.1109/HPCSim.2012.6266902.
15. E. J. Hughes. Many-objective directed evolutionary line search. In Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, GECCO '11, pages 761–768, New York, NY, USA, 2011. ACM.
16. Kalyanmoy, Deb. Multi-Objective Optimization Using Evolutionary Algorithms, pages 124-124, 2001.
17. D. A. V. Veldhuizen and G. B. Lamont. Evolutionary computation and convergence to a Pareto front. In Stanford University, California, pages 221–228. Morgan Kaufmann, 1998.
18. A. Zhou, Y. Jin, Q. Zhang, B. Sendhoff, and E. Tsang. Combining model-based and genetics-based offspring generation for multi-objective optimization using a convergence criterion. In Evolutionary Computation, 2006. CEC 2006. IEEE Congress on, pages 892–899, 2006.
19. D. A. V. Veldhuizen and G. Lamont. On measuring multiobjective evolutionary algorithm performance. In 2000 Congress on Evolutionary Computation, pages 204–211,. 2000.
20. E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. Evol. Comput., 8(2):173–195, June 2000.
21. K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable test problems for evolutionary multiobjective optimization. In A. Abraham, L. Jain, and R. Goldberg, editors, Evolutionary Multiobjective Optimization, Advanced Information and Knowledge Processing, pages 105–145. Springer London, 2005.
22. Marler, R.T. and Arora, J.S.. Survey of multi-objective optimization methods for engineering. Structural and Multidisciplinary Optimization, volume 26 pages 369-395, April 2004.
23. J.J. Durillo and A.J. Nebro and E. Alba. The jMetal Framework for Multi-Objective Optimization: Design and Architecture. In Evolutionary Computing. CEC 2010. IEEE Congress on, pages 4138-4325, 2010.