Evolutionary Approach to Multiobjective Optimization of Portfolios that Reflect the Behaviour of Investment Funds

Krzysztof Michalak[1], Patryk Filipiak[2], and Piotr Lipinski[2]

[1] Institute of Business Informatics,
Wroclaw University of Economics, Wroclaw, Poland
`krzysztof.michalak@ue.wroc.pl`
[2] Institute of Computer Science,
University of Wroclaw, Wroclaw, Poland
`{patryk.filipiak,lipinski}@ii.uni.wroc.pl`

Abstract. This paper addresses a problem of finding portfolios that perform better than investment funds while showing similar behaviour. The quality of investment portfolio can be measured using various criteria such as the return and some kind of risk measurement. Investors seek to maximize return while minimizing risk. In order to achieve this goal various instruments are considered. One of the possibilities is to entrust the assets to an investment fund. Investment funds build their own portfolios of stocks, bonds, commodities, currencies, etc.
In this paper we consider the problem of finding a portfolio which outperforms a given investment fund with respect to both the return and the risk and which also behave in a similar way to the given fund. The rationale behind such an approach is that investment strategies of mutual funds are prepared by experts and are therefore expected to be reasonably good in terms of both the return and the risk. To achieve the presented goal we use a multiobjective evolutionary algorithm with a dedicated "division mutation" operator and a local search procedure. Presented method seems capable of building portfolios with desired qualities.

Keywords: investment funds, portfolio optimization, multiobjective evolutionary optimization

1 Introduction

Bio-inspired methods are often applied to various economic and financial problems [1, 5, 15] such as supporting financial decision making [10, 14, 17] and discovering trading rules for stock market speculations [4, 11, 12]. Optimization of investment portfolios [2] is an important task with practical implications to which evolutionary methods are often employed [13, 16]. Usually, there is more than one criterion to optimize. Typically the return and at least one measure of investment risk are considered. In this paper we focused on finding portfolios of investment instruments that not only maximize the return and minimize the risk but also behave similarly to a given investment fund. Performing such search is motivated by the fact, that we would like to know if in the "neighbourhood" of an investment fund strategy there exist other strategies that can get better results.

In this paper we assume that we have a number of categories C_j, $j = 1, \ldots, N^{(c)}$ of investment instruments, such as stocks, bonds, currencies, etc. Each of the considered instruments I_i, $i = 1, \ldots, N^{(i)}$ belongs to exactly one instrument category C_j. Therefore, obviously, $N^{(i)} = \sum_{j=1}^{N^{(c)}} |C_j|$. We denote the quotations of instrument I_i by $q_i(t)$.

The portfolio is represented as a vector $w \in R^{N^{(i)}}$ in which each coordinate w_i is a "weight" of instrument I_i (the number of units of this instrument bought when the investment is made). We calculate the quotations $p(t)$ of portfolio w as:

$$p(t) = \sum_{i=1}^{N^{(i)}} w_i q_i(t) \ . \tag{1}$$

The quotations for the instruments (and therefore for the entire portfolio as well) are assumed to be available for a time period $[t_{min}, t_{max}]$.

There are three criteria which are optimized in this paper: the return measure, the risk measure and the dissimilarity to a given investment fund. The return R is assumed to be the ratio between the last and the first quotation, so $R_f = f(t_{max})/f(t_{min})$ for the investment fund, $R_{I_i} = q_i(t_{max})/q_i(t_{min})$ for instruments and $R_w = p(t_{max})/p(t_{min})$ for portfolios. As this is essentially a return calculation for a buy-and-hold strategy, trading costs have similar impact on the return of all portfolios and are therefore not taken into consideration in this study. The risk for the investment fund (V_f), the instruments (V_{I_i}) and portfolios (V_w) is measured using the classical measure of Value at Risk [9] calculated for the entire period $[t_{min}, t_{max}]$. The dissimilarity $D_{w,f}$ between portfolio w quotations $p(t)$ and investment fund quotations $f(t)$ is measured using Mean Squared Error (MSE) measure calculated for the entire period $[t_{min}, t_{max}]$. We intend to find portfolios which maximize the return R_w, and minimize both the risk V_w and dissimilarity $D_{w,f}$ for a given investment fund f.

2 Proposed method

Finding portfolios which maximize the return R_w, and minimize both the risk V_w and dissimilarity $D_{w,f}$ for a given investment fund f requires three-objective optimization. It is well-known that multiobjective optimization can be performed effectively using evolutionary methods [6, 18]. One of the reasons for which such methods are particularly useful in this context is that population of solutions can readily be used to represent a Pareto set and a Pareto front of a sovled problem. In this paper we propose an evolutionary algorithm described below. The algorithm keeps a subpopulation P_{C_j} of specimens for each category C_j,

$j = 1, \ldots, N^{(c)}$ of investment instruments and one global population P_{all} which, apart from evolving on its own, is also updated by importing specimens from other subpopulations. For a given j, specimens in subpopulation P_{C_j} have non-zero weights assigned only to instruments that belong to category C_j (eg. stocks). All genetic operations and other procedures are implemented to ensure that this condition is met during entire program execution. Algorithm 1 presents details of the evolutionary algorithm used for performing portfolio optimization. In the algorithm $P_{all}(g)$ and $P_{C_j}(g)$ denote global and category-specific subpopulations at the generation g.
The following quantities parameterize the algorithm:

$N^{(gen)}$ - the number of generations,

$N^{(pop)}$ - initial size of each subpopulation (global and category-specific),

$N^{(cross)}$ - the number of crossover operations per generation,

$P^{(mut)}$ - the percentage of individuals that produce offspring using mutation,

$P^{(weight\_mut)}$ - the percentage of weights mutated in one specimen.

$P^{(div\_mut)}$ - the percentage of individuals that produce offspring using the dedicated "division mutation" operator,

Algorithm 1 Evolutionary algorithm with subpopulations for optimization of portfolios.

```
$P_{all}(1) = InitPopulation(N^{(pop)})$
**for** $j = 1 \to N^{(c)}$ **do**
  $P_{C_j}(1) = InitPopulation(N^{(pop)})$
**end for**
**for** $g = 1 \to N^{(gen)}$ **do**
  **for** $j = 1 \to N^{(c)}$ **do**
    $P_{C_j}(g) = P_{C_j}(g) \cup Crossover(P_{C_j}(g))$
    $P_{C_j}(g) = P_{C_j}(g) \cup Mutate(P_{C_j}(g))$
    $P_{C_j}(g) = P_{C_j}(g) \cup DivMutate(P_{C_j}(g))$
    $P_{C_j}(g) = P_{C_j}(g) \cup LocalSearch(P_{C_j}(g))$
  **end for**
  **for** $j = 1 \to N^{(c)}$ **do**
    $P_{all}(g) = P_{all}(g) \cup P_{C_j}(g)$
  **end for**
  $P_{all}(g) = P_{all}(g) \cup Crossover(P_{C_j}(g))$
  $P_{all}(g) = P_{all}(g) \cup Mutate(P_{C_j}(g))$
  $P_{all}(g) = P_{all}(g) \cup DivMutate(P_{C_j}(g))$
  $P_{all}(g) = P_{all}(g) \cup LocalSearch(P_{C_j}(g))$
  $P_{all}(g+1) = Select(P_{all}(g))$
  **for** $j = 1 \to N^{(c)}$ **do**
    $P_{C_j}(g+1) = Select(P_{C_j}(g))$
  **end for**
**end for**
```

The algorithm uses the following procedures and operators:

InitPopulation - creates a new population consisting of a given number of specimens. Each specimen is a vector $w \in R^{N^{(i)}}$ in which each coordinate w_i is drawn with uniform probability from the range $[0, w_i^{(max)}]$, where:

$$w_i^{(max)} = \frac{\max_{t \in [t_{min}, t_{max}]} f(t)}{\min_{t \in [t_{min}, t_{max}]} I_i(t)} \ . \tag{2}$$

Crossover - a standard single-point crossover operator [7]. $2N^{(cross)}$ new specimens are generated by selecting two parents at random from existing population and then splicing the chromosomes at a randomly selected point. These new specimens are then added to existing population.

Mutate - creates $[P^{(mut)} \cdot \text{population\_size}/100]$ new specimens. Each new specimen is generated by selecting at random from existing population one specimen in which $P^{(weight\_mut)}$ percent of weights are mutated. A single weight is mutated by selecting an integer exponent η with uniform probability from range $[-6, 6]$, a real number increment δ with uniform probability from range $[-5, 5]$ and then adding $\eta \cdot \delta$ to the weight. The new value is then clipped to the range $[0, w_i^{(max)}]$, where $w_i^{(max)}$ is given by the Equation (2).

DivMutate - a dedicated mutation operator described below.

LocalSearch - a local search procedure described below.

Select - selection procedure based on non-domination sorting and crowding distance known from the NSGA-II algorithm [3]. This selection scheme involves performing a series of binary tournaments in which the fitter of two individuals is selected as a winner. Specimen fitness is not directly calculated, but rather the selection is based on the ranks of Pareto fronts to which the specimens belong and the density of population near each of the specimens. In addition to specimens selected in binary tournaments, a set of specimens that outperform the given investment fund with respect to both the return and the risk is copied from one generation to the next. This set may contain at most $N^{(elite)}$ specimens with the largest crowding distance that are non-dominated by each other (dominated ones are removed).

2.1 Dedicated "division mutation" operator

During preliminary experiments we noticed, that if the initial portfolio time series differs significantly from the series of quotations of the investment fund the convergence is very slow. We hypothesized, that this is caused by the fact that specimens are evaluated using three criteria, so even specimens relatively distant from the target time series may survive. We also suspected, that the simple mutation operator described above may not be very effective in generating new specimens that lie far from the original ones. This could be remedied by increasing the probability of mutations or the range of possible weight change.

Unfortunately, such approach would cause abrupt changes in the structure of portfolios, which we would like to avoid. Therefore, we proposed a new genetic operator which acts as mutation and is specific to the problem we tackle.

"Division mutation" operator creates $[P^{(div\_mut)} \cdot \text{population\_size}/100]$ new specimens. Each new specimen is generated by selecting at random from existing population one specimen and modifying it according to the Algorithm 2.

Algorithm 2 Division mutation for a single specimen $w \in R^{N^{(i)}}$.

```
Select at random $t_0 \in [t_{min}, t_{max}]$
$\delta = p(t_0)/f(t_0)$
for $j = 1 \to N^{(i)}$ do
    $w_i = w_i/\delta$
end for
```

From the Equation (1) it is obvious that after the application of this operator the new quotations $p'(t)$ of the portfolio satisfy a condition: $\forall_t \cdot p'(t) = p(t)/\delta$, where $p(t)$ are the quotations before the application of division mutation. Therefore $p'(t_0) = p(t_0)/\delta = p(t_0)/(p(t_0)/f(t_0)) = f(t_0)$, which means, that the "division mutation" operator ensures that the time series of portfolio quotations intersects with the time series of fund quotations at a randomly selected point. This operator however, scales all the weights by the same factor and thus it leaves the structure of the portfolio intact as opposed to the regular mutation which works on individual weights.

2.2 Local search

Local search is a well-known approach to improving the quality of solutions generated by an evolutionary algorithm [8]. In this paper we introduce a local search procedure which tries to find a solution that improves two objectives by combining solutions that are good with respect to one of the objectives.

In the case of the particular problem of finding portfolios of investment instruments we try to find portfolios that improve both the return and the risk by combining portfolios that give higher returns than a given investment fund or have a lower risk. The local search procedure operates as shown in Algorithm 3. It selects two random samples each containing $N^{(ls)}$ specimens which are better than the given investment fund with respect to one of the objectives. Then, linear combinations of all pairs of specimens from the two samples are generated. New specimens for which both objectives are better than threshold values θ_1 and θ_2 are added to the result set. In this paper we set $\theta_1 = R_f$ and $\theta_2 = V_f$, i.e., specimens that have both higher return and lower risk that the given fund are added to the result set.

If any specimens that improve both the return and the risk are found during local search they are given a chance to produce offspring using division mutation

Algorithm 3 Local search procedure.

```
$A = \emptyset$ - a set of specimens found by the local search
$S_1$ = a random sample of $N^{(ls)}$ specimens with the objective $O_1$ better than $\theta_1$
$S_2$ = a random sample of $N^{(ls)}$ specimens with the objective $O_2$ better than $\theta_2$
for $i = 1 \to N^{(ls)}$ do
  for $j = 1 \to N^{(ls)}$ do
    $w_i$ = weights of specimen $S_1[i]$
    $w_j$ = weights of specimen $S_2[j]$
    for $\alpha = 0.01 \to 1$ step 0.01 do
      $w' = \alpha w_i + (1 - \alpha) w_j$
      $O'_1$ = objective 1 obtained from $w'$
      $O'_2$ = objective 2 obtained from $w'$
      if $O'_1$ is better than $\theta_1$ and $O'_2$ is better than $\theta_2$ then
        $A = A \cup \{$ a new specimen with weights $w'\}$
      end if
    end for
  end for
end for
$A = A \cup DivMutate(A)$
$A = SelectNonDominated(A)$
return $A$
```

operator. Finally, from the result set A only these specimens are selected that are not dominated by any specimens in A.

3 Experiments

In the experiments we tested the ability of the presented algorithm to construct portfolios of stocks and currencies that outperform a given investment fund with respect to both the return and the risk at the same time. We performed tests for 18 Polish investment funds from which some invest mainly in stocks and other invest mainly in instruments having a lower risk such as bonds. Details of the funds are summarized in Table 1. The available investment instruments were 317 stocks from Warsaw Stock Exchange and 6 currencies that have exchange rates to Polish currency noted on FOREX market: Swiss franc (CHF), Czech koruna (CZK), Euro (EUR), British pound (GBP), Hungarian forint (HUF) and US dollar (USD). We assumed that the investment is made for a period of half a year from $t_{min} = 2010.06.30$ to $t_{max} = 2010.12.31$. This period equals the interval between semi-annual reports that investment funds are required to publish from which we could learn about the structure of portfolio of instruments used by each of the funds.

In the experiments we ran the algorithm for $N^{(gen)} = 50$ generations. As there were two categories of investment instruments (stocks and currencies) the algorithm kept three subpopulations (one global and two category-specific) which initially consisted of $N^{(pop)} = 200$ specimens. The other parameters of the evo-

Table 1. Details of investment funds considered in the paper

ID	Issuer	Main investment instrument (as reported on 2010.06.30)	Return on $[t_{min}, t_{max}]$	VaR on $[t_{min}, t_{max}]$
96	Millenium	stocks (92.7%)	1.193616	0.009697616
97	Millenium	stocks (59.6%)	1.126392	0.007921864
107	Commercial Union	debt securities (97%)	1.020221	0.0005879808
108	Commercial Union	debt securities (89%)	1.021743	0.00259334
109	Commercial Union	stocks (94%)	1.231499	0.01066058
121	Commercial Union	debt securities (62%)	1.098829	0.003664982
149	Millenium	debt securities (63,6%)	1.081201	0.004770102
159	Commercial Union	debt securities (63%)	1.075789	0.001771456
261	Commercial Union	stocks (51%)	1.13904	0.005648553
1619	Commercial Union	investment funds (50%)	1.056212	0.006571422
2378	Commercial Union	stocks (96%)	1.248263	0.01144357
2379	Commercial Union	stocks (90%)	1.288662	0.006936816
2380	Commercial Union	stocks (80%)	1.250754	0.007484911
2381	Commercial Union	stocks (92%)	1.279447	0.007234238
3463	Millenium	investment funds (81.5%)	0.972389	0.009257582
3466	Millenium	investment funds (86.7%)	1.036602	0.01081419
3467	Millenium	investment funds (80.8%)	0.8973272	0.01727394
3470	Millenium	investment funds (88%)	1.016006	0.01577246

lutionary algorithm were set to $N^{(cross)} = 50$, $P^{(mut)} = 200$, $P^{(weight\_mut)} = 50$, $P^{(div\_mut)} = 10$, $N^{(elite)} = 100$. Local search sample size was set to $N^{(ls)} = 20$.

Table 2 summarizes the experiments. This table presents, for each investment fund, the number of generation at which the first portfolio with higher return and lower risk than the given fund was found.

Table 2. Number of generation at which the first portfolio with higher return and lower risk than the given investment fund was found

ID	96	97	107*	108*	109	121*	149*	159*	261	1619	2378	2379	2380	2381	3463	3466	3467	3470
Generation Pop_{all}	1	1	-	-	1	-	-	-	39	2	1	16	5	3	1	1	1	1
Generation Pop_{stocks}	2	12	-	-	2	-	-	-	-	-	3	-	-	-	1	1	1	1
Generation $Pop_{currencies}$	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-

*Higher return portfolios have been found, but not lower risk portfolios.

As can be seen from the results we were able to find portfolios of both stocks and currencies that outperform investment funds which invest mainly in stocks or other investment funds. It was not possible to outperform funds that invest in debt securities. A comments under Table 2 along with a glance at VaR values in Table 1 explain why. The funds that invest in debt securities have very low risk, which is not surprising as debt securities are known to possess exactly that feature. The algorithm was able to construct portfolios that give higher return, but at the expense of increasing the risk. It is also worth noticing, that the algorithm had not have any low-risk instruments at its disposal (only stocks and currencies were available).

It seems that in some cases the presence of currencies in the investment portfolio was required to outperform the given investment fund. Only in one case however a portfolio containing only currencies was enough to ensure enough return and sufficiently low risk.

In order to visualize the behaviour of the algorithm we include graphs that present minimum, average and maximum values of the three objective throughout all 50 generations of evolution for calculations performed for one of the funds ($ID = 96$). Due to space limitations it is not possible to present similar graphs for all investment funds considered in the tests. Top-left graph in Figure 3 shows values of the return obtained from investment in a given portfolio. Maximum values in this graph reach higher than 1.5, while the return of the fund is less than 1.2, but this might have been achieved at the expense of increasing the risk (which is not shown in this particular graph). In top-right graph in Figure 3 values of Value-at-Risk measure are shown. Again, risk given by individual specimens is more than two times lower than that of the fund, but it might have been achieved at the expense of lower return. Bottom-left graph in Figure 3 presents values of Mean Squared Error calculated on the range $[t_{min}, t_{max}]$ (2010.06.30-2010.12.31) between portfolio quotations $p(t)$ and investment fund quotations $f(t)$. Due to the large changes in the value of the MSE especially at the beginning of the evolution this graph has a logarithmic scale.

As it can be seen from the graphs in the case of this particular fund the algorithm gradually improves the return for about 35 generations and the risk for about 25 generations. A significant pressure is visible with respect to similarity to the investment fund behaviour.

4 Conclusion

In this paper we address a problem of finding portfolios of investment instruments that give higher return and achieve lower risk than a given investment fund. An additional criterion that was expected to be optimized was that the constructed investment strategy should behave similarly to the investment fund. We proposed an multiobjective evolutionary algorithm that optimized the solutions taking into consideration three objectives: investment return, Value-at-Risk as a risk measure and Mean Squared Error as a measure of dissimilarity between the time series of portfolio quotations and the time series of quotations of the given investment fund. In the paper we introduce a new mutation operator aimed at increasing convergence to the baseline time series and a local search procedure which combines solutions which are superior with respect to at least one objective (risk or return) in order to find solutions superior with respect to two criteria. During the tests we found out that the new mutation operator speeds

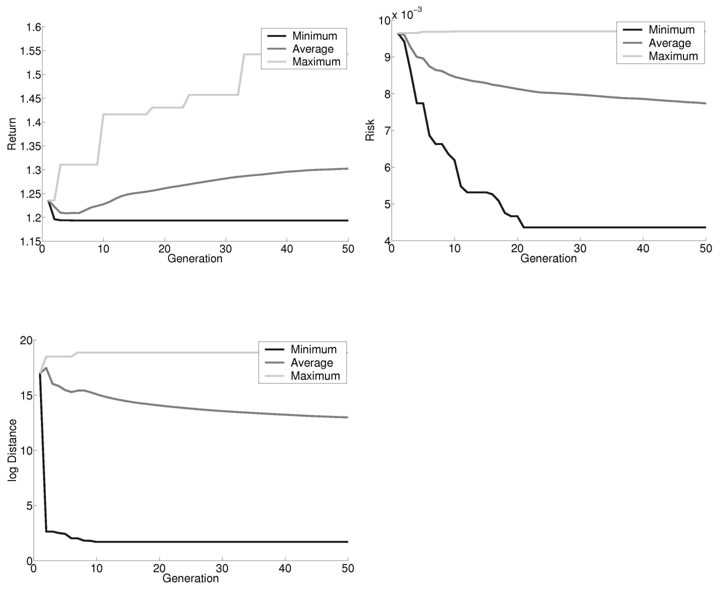

Fig. 1. Minimum, average and maximum values of return (top left), risk (top right) and the logarithm of MSE measuring dissimilarity between fund and portfolio quotations (bottom left) in the population in each of the generations

up convergence to the given investment fund time series by generating specimens that intersect with the fund quotations at at least one instant in time.

The experiments were performed using stocks and currencies as possible investment instruments. In the experiments portfolios that outperform investment funds were found in cases when the fund invests mainly in stocks or other investment funds. In the case when the fund invests mainly in debt securities finding portfolios outperforming the fund in minimizing the risk was not possible. This is most probably due to inherently higher risk of stock and currency investments compared to debt securities which are known to have lower investment risk.

In general, however, the proposed algorithm seems to be capable of finding portfolios with required properties, i.e. similar to a given investment fund but with higher return and lower risk, provided that proper investment instruments are available for portfolio construction.

References

1. Bauer, R.: Genetic Algorithms and Investment Strategies. Wiley, Chichester (1994)
2. Best, M. J.: Portfolio Optimization. Chapman&Hall/CRC (2010)
3. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. In: PPSN VI, LNCS, vol. 1917, pp. 849–858, Springer, Heidelberg (2000)

4. Dempsey, I., O'Neill, M., Brabazon, A.: Adaptive Trading with Grammatical Evolution. In: Proceedings of the 2006 Congress on Evolutionary Computation (CEC 2006), pp. 2587–2592, IEEE, Los Alamitos (2006)
5. Dempster, M., Jones, C.: A Real-Time Adaptive Trading System using Genetic Programming. Quantitative Finance 1, pp. 397–413, (2001)
6. Filipiak, P, Michalak, K., Lipinski, P.: Infeasibility Driven Evolutionary Algorithm with ARIMA-Based Prediction Mechanism. LNCS, vol. 6936, pp. 345–352, Springer (2011)
7. Haupt, R. L., Werner, D. H.: Genetic Algorithms in Electromagnetics. pp. 37–38, John Wiley & Sons, Inc. (2007)
8. Jaszkiewicz, A.: Genetic local search for multiobjective combinatorial optimization. European J. of Operational Research. 137, pp. 50–71 (2002)
9. Jorion, P.: Value at Risk: The New Benchmark for Managing Financial Risk. McGraw-Hill (2006)
10. Li, J., Taiwo, S.: Enhancing Financial Decision Making Using Multi-Objective Financial Genetic Programming. In: Proceedings of IEEE Congress on Evolutionary Computation, 2006, pp. 2171–2178, (2006)
11. Lipinski, P.: Dependency Mining in Large Sets of Stock Market Trading Rules. In: Enhanced Methods in Computer Security, Biometric and Intelligent Systems, ed. J. Pejas, A. Piegat, pp. 329–336, Kluwer Academic Publishers (2005)
12. Lipinski, P.: Discovering Stock Market Trading Rules using Multi-Layer Perceptrons. In: Proceedings of 9th International Work Conference on Artificial Neural Networks, IWANN 2007 LNCS, vol. 4507, pp. 1114–1121, Springer (2007)
13. Lipinski, P.: Evolutionary Strategies for Building Risk-Optimal Portfolios. In: Natural Computing in Computational Finance, SCI, vol. 100, pp. 53–65, Springer (2008)
14. Lipinski, P.: Frequent Knowledge Patterns in Evolutionary Decision Support Systems for Financial Time Series Analysis. In: Natural Computing in Computational Finance, SCI, vol. 293, pp. 131–145, Springer (2010)
15. Michalak, K., Lipinski, P.: Prediction of high increases in stock prices using neural networks. Neural Network World, 15, vol. 4, pp. 359-366, Springer (2005)
16. Radziukyniene, I, Zilinskas, A.: Evolutionary Methods for Multi-Objective Portfolio Optimization. In: Proceedings of the World Congress on Engineering 2008, pp. 1155–1159, Newswood Limited (2008)
17. Tsang, E., Li, J., Markose, S., Er, H., Salhi, A., Iori, G.: EDDIE in Financial Decision Making. Journal of Management and Economics 4(4), (2000)
18. Zitzler, E., Thiele, L.: Multiobjective optimization using evolutionary algorithms – A comparative case study. In: Parallel Problem Solving from Nature – PPSN V, LNCS, vol. 1498, pp. 292–301. Springer, Berlin / Heidelberg (1998)