A performance comparison of multi-objective optimization algorithms for solving nearly-zero-energy-building design problems

Citation for published version (APA):

DOI:
10.1016/j.enbuild.2016.03.035

Document status and date:
Published: 01/06/2016

Document Version:
Accepted manuscript including changes made at the peer-review stage

Please check the document version of this publication:

• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.
A performance comparison of multi-objective optimization algorithms for solving nearly-zero-energy-building design problems

Mohamed Hamdy\textsuperscript{1,2,3}, Anh-Tuan Nguyen\textsuperscript{4}, Jan L. M. Hensen\textsuperscript{1}

\textsuperscript{1} Department of the Built Environment, Building Physics and Services, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands.
\textsuperscript{2} Department of Mechanical Power Engineering, Helwan University, P.O. Box 11718 Cairo, Egypt.
\textsuperscript{3} Department of Energy Technology, Aalto University School of Engineering, P.O. Box 14400, FI-00076 Aalto, Finland.
\textsuperscript{4} Faculty of Architecture, The University of Danang - University of Science and Technology, 54 Nguyen Luong Bang, Danang, Vietnam.

ABSTRACT

Integrated building design is inherently a multi-objective optimization problem where two or more conflicting objectives must be minimized and/or maximized concurrently. Many multi-objective optimization algorithms have been developed; however few of them are tested in solving building design problems.

This paper compares performance of seven commonly-used multi-objective evolutionary optimization algorithms in solving the design problem of a nearly zero energy building (nZEB) where more than 1.6\textsuperscript{10} solutions would be possible. The compared algorithms include a controlled non-dominated sorting genetic algorithm with a passive archive (pNSGA-II), a multi-objective particle swarm optimization (MOPSO), a two-phase optimization using the genetic algorithm (PR_GA), an elitist non-dominated sorting evolution strategy (ENSES), a multi-objective evolutionary algorithm based on the concept of epsilon dominance (evMOGA), a multi-objective differential evolution algorithm (spMODE-II), and a multi-objective dragonfly algorithm (MODA). Several criteria was used to compare performance of these algorithms.

In most cases, the quality of the obtained solutions was improved when the number of generations was increased. The optimization results of running each algorithm 20 times with gradually increasing number of evaluations indicated that the PR_GA algorithm had a high repeatability to explore a large area of the solution-space and achieved close-to-optimal solutions with a good diversity, followed by the pNSGA-II, evMOGA and spMODE-II. Uncompetitive results were achieved by the ENSES, MOPSO and MODA in most running cases. The study also found that 1400 - 1800 were minimum required number of evaluations to stabilize optimization results of the building energy model.

Keywords: multi-objective optimization; algorithms; experimentation; building simulation; comparison
Contents

ABSTRACT 1

Contents 2

1. Background and objectives of the study 3

2. An overview on the performance of optimization algorithms in building energy analysis 4
   2.1. Performance comparison of single objective optimization algorithms 5
   2.2. Performance comparison of MOOAs 6

3. Research methodology 7
   3.1. About the selection of test algorithms 7
   3.2. Test models 9
   3.3. Objectives to be minimized 11
   3.4. Criteria for performance comparison 12

4. Performance of the optimization algorithms 16
   4.1. Convergence to the best Pareto front and diversity of obtained solutions 17
   4.2. Execution time 21
   4.3. Number of solutions on the Pareto-optimal set 21
   4.4. Performance ranking and some observations 23

5. Discussion and conclusion 25

6. Appendix 26
   6.1. Non-dominated Sorting Genetic Algorithm II with a passive archive (pNSGA-II) 26
   6.2. Multi-objective particle swarm optimization 27
   6.3. Two-phase optimization using Genetic algorithm (PR_GA) 27
   6.4. Multi-objective evolutionary algorithm based on the concept of epsilon dominance (evMOGA) 28
   6.5. Elitist non-dominated sorting evolution strategy (ENSES) 28
   6.6. Multi-objective differential evolution algorithm with spherical pruning based on preferences – an improved version of the spMODE algorithm (spMODE-II) 28
1. Background and objectives of the study

Today, simulation-based optimization has become an efficient design approach to satisfy several stringent requirements in designing high performance buildings (e.g. low-energy buildings, passive houses, green buildings, net zero-energy buildings, zero-carbon buildings…) [1]. In real-world building design problems designers often have to deal with conflict design criteria simultaneously such as minimum energy consumption versus...
maximum thermal comfort, minimum energy consumption versus minimum construction cost… Multi-objective optimization is therefore, in many cases, more relevant than the single-objective approach. This has led to the application of multi-objective optimization algorithms (MOOAs) that identify the Pareto optimum trade-off between conflicting design objectives (e.g. [2, 3, 4, 5, 6]).

It is found that efficient MOOAs are essential to find the optimal solutions without a need for numerous time consuming simulations. However, performance of MOOAs on building optimization problems has not been well understood due to the lack of research-based evidences. This study is among the first efforts that comprehensively investigate performance of seven evolutionary optimization algorithms in solving multi-objective optimization problems by using the simulation-based optimization approach. The test problem is a building energy model, which has a discrete solution space of energy saving measures and energy supply systems options, including renewable energy sources (RES). The major aims of this study are:

- To compare performance of different evolutionary algorithms in optimizing building energy models;
- To understand behavior of these evolutionary algorithms in solving a multi-objective optimization problem.

In this study, we provide an overview on performance comparison of optimization algorithms in building energy analysis. The investigated MOOAs and their performance criteria are described in the next section, which is followed by a description of the nZEB optimization problem (that is in accordance with the implementation of the new EPBD-2010). The results of this study were reported to support the performance criteria.

2. An overview on the performance of optimization algorithms in building energy analysis

Due to the large amount of design variables of building energy models as well as their discrete, non-linear, and highly constrained characteristics, simulation results are generally multi-modal and discontinuous, generating discontinuities or noise in the objective functions in building optimization problems (BOPs). As a result, optimization algorithms that require smoothness were found not efficient [7, 8]. In many cases, stochastic population-based MOOAs (evolutionary optimization, swarm intelligence...) that do not require smoothness are able to handle the discontinuity of the search space. It is clear that performances of different algorithms are not equal. An algorithm may perform well by one criterion but fails by other criteria. Thus, performance of the optimization algorithms in solving BOPs is really an attractive research question that needs to be investigated.
2.1. Performance comparison of single objective optimization algorithms

The performance is often considered the major criterion for selecting an optimization algorithm. There have been several studies on the performance of different optimization algorithms on single-objective BOPs. Wetter and Wright [9] compared the performance of a Hooke-Jeeves algorithm and a GA in optimizing building energy consumption. Their result indicated that the GA outperformed the Hooke-Jeeves algorithm and the latter was attracted by a local minimum. Wetter and Wright [7] compared the performance of eight algorithms (Coordinate search algorithm, Hooke-Jeeves algorithm, particle swarm optimization (PSO), PSO that searches on a mesh, hybrid PSO-Hooke-Jeeves algorithm, simple GA, simplex algorithm of Nelder and Mead, discrete Armijo gradient algorithm) in solving simple and complex building models using a low number of cost function evaluations. Performance criteria include number of iterations, and optimal objective values. They found that the GA consistently got close to the best minimum and the Hybrid algorithm achieved the overall best cost reductions (although with a higher number of simulations than the simple GA). When the discontinuities in the cost function were small, the Hooke–Jeeves algorithm achieved good performance with a small number of iterations. Performances of other algorithms were not stable and the use of simplex algorithm and discrete Armijo gradient algorithm were not recommended. In GenOpt [10], Wetter introduced an improved hybrid algorithm PSO - Hooke-Jeeves in which the PSO performs the search on a mesh, significantly reducing the number of function evaluations called by the algorithm. Kampf et al. [11] examined the performance of two hybrid algorithms (PSO - Hooke-Jeeves and CMA-ES/HDE) in optimizing five standard benchmark functions (Ackley, Rastrigin, Rosenbrock, sphere functions and a highly-constrained function) and real-world problems using EnergyPlus simulation program. The results indicated that the CMA-ES/HDE performed better than the PSO - Hooke-Jeeves in solving the benchmark functions with 10 dimensions or less. However, if the number of dimensions is larger than 10, the hybrid PSO - Hooke-Jeeves gave better solutions. Both these algorithms performed well with the real-world BOPs using EnergyPlus models. Lee et al. [12] compared the performance of the differential evolution algorithm with that of a PSO, a GA and the Lagrangian method by solving the optimal chiller loading problem for reducing energy consumption. They found that the proposed differential evolution algorithm could give similar results as the PSO did, but obtained better average solutions. The differential evolution algorithm outperformed the GA in finding optimal solutions and also overcame the divergence problem caused by the Lagrangian method occurring at low demands.

Tuhus-Dubrow and Krarti [13] examined the performance of a GA against the PSO and the sequential search method in building envelope design to minimize lifecycle cost. The result reveals that the GA was more efficient than the two remaining methods in the complex cases (more than 10 optimization parameters were included in the building model). Moreover, the GA optimization method could define the optimal solution with an accuracy
of 0.5%, and required only a half of the number of iterations needed by the PSO and the sequential search method.

Bichiou and Krarti [14] compared a GA with the PSO and the sequential search in optimizing building envelops and HVAC system design, in terms of computational time and cost reduction. They found that the GA and the PSO required typically less computational time to obtain optimal solutions than the sequential search. The optimal results given by the three algorithms were almost similar.

From these studies, it can be seen that the stochastic population-based optimization algorithms (e.g. evolutionary algorithms, swarm intelligence…) generally outperform the others and are likely suitable for BOPs. However, it is worthy of note that the cost reduction by an algorithm not only depends on the natures of the algorithm, but also depends on the settings of algorithm parameters [7, 11]. According to the so-called ‘no free lunch theorem’ [15], there is no single best algorithm for all optimization problems. Hence, algorithm selection and settings might involve trial and error [1].

2.2. Performance comparison of MOOAs

In contrary to the case of single-objective optimization algorithm, performance of MOOAs on BOPs has not been well understood due to the lack of research-based evidences. Nassif et al. [16] compared the NSGA and the NSGA-II in solving simplified variable air volume HVAC design problems. The comparison criteria includes the distance to the true Pareto front and the spread of optimal solutions. They went to conclusion that the NSGA-II performed better than the NSGA, both in terms of distance and spread.

Hofpe [17] compared performance of two MOOAs: the NSGA-II and the SMS EMOA, by comparing their solutions on the Pareto fronts. She found that the performance of the NSGA-II was not satisfied in all tests, while the SMS EMOA gave more competitive results but the number of simulations was much higher. The failure of the NSGA-II was still unknown.

Brownlee et al. [18] compared the performance of five multi-objective algorithms (IBEA, MOCell, NSGA-II, SPEA2 and PAES) along with the random search method in solving a multi-objective problem concerning window placement. The assessment criteria included the size of hypervolume and the spread established by the Pareto front found after 5000 evaluations. They found that the NSGA-II outperformed the others, both in terms of the size of hypervolume and the spread of optimal solutions.

Hamdy et al. [2] evaluated performance of three multi-objective algorithms, including the NSGA-II, aNSGA-II (NSGA-II with active archive) and pNSGA-II (NSGA-II with a passive archive strategy), on a BOP and 2 benchmark test problems. They reported that the aNSGA-II has a better repeatability in finding high-quality solutions close to the true Pareto front with fewer evaluations and high convergence than the original NSGA-II and pNSGA-II.
Hamdy et al. [19] proposed two new approaches in HVAC design optimization by using combination of different algorithms. They compared performance of the original genetic algorithm implemented in Matlab with the “so called” PR_GA (Running the Fmincon solver and Fminimax function in Matlab optimization toolbox to prepare a good initial population for the first generation of the GA) and the GA_RF (a combination between the genetic algorithm in Matlab and the sequential quadratic programming (SQP) method to improve or refine some of GA Pareto points as well as to enhance the Pareto front with additional refined solutions). For nearly equal number of simulation runs, Hamdy et al. found that the PR_GA and the GA_RF approaches provided much better Pareto fronts and contributed more Pareto solutions than the original GA while maintaining reasonable computational time (less than the GA). The PR_GA approach was included in the test of this study.

Multi-objective evolutionary algorithms (MOEAs) were used for several purposes, e.g. in designing cost-effective environmentally-friendly buildings [4, 20], low-energy highly-comfort buildings [3] or improving the energy consumption of net-zero energy solar homes [21]. Elitist MOEAs (e.g., NSGA-II) offer great potential for implementations in this field. However and because of their stochastic operators, they could occasionally fail to get close to the Pareto-optimal front, particularly if a low number of iterations is implemented as the stopping criterion [22, 23, 6]. Researchers, therefore, need more detailed guidelines more comprehensive comparative tests so as to fully understand the natures of MOOAs in solving BOPs. Also, an advanced test method is essential to fully examine the performance of these algorithms. This is also the major objective of this paper.

3. Research methodology

3.1. Selection of test algorithms

A comprehensive review on optimization methods applied to BOPs has been done by the authors [1]. The result indicates that the evolutionary algorithms and the swarm intelligence algorithms were the most commonly-used methods (see Figure 1). In another study [24], it was found that evolutionary algorithms (GA, NSGA-II, swarm intelligence algorithms…) were among the most popular optimization methods. These findings allow us to identify the potential algorithms to be investigated in this study.
Testing all evolutionary algorithms is well beyond the scope of this paper because of a large number of existing evolutionary optimization methods. After considering all possibilities, 7 popular MOOAs were selected for the test, including: a controlled non-dominated sorting genetic algorithm with a passive archive (a variant of the NSGA-II), a multi-objective particle swarm optimization (MOPSO), a two-phase optimization GA (PR_GA), a multi-objective evolutionary algorithm based on the concept of epsilon dominance (evMOGA), an elitist non-dominated sorting evolution strategy (ENSES), a multi-objective differential evolution algorithm (spMODE-II), and a multi-objective dragonfly algorithm (MODA). These algorithms were developed along 2002 to 2014. Their codes were exploited from MathWorks/from the authors and then implemented into Matlab optimization toolbox for use. The natures of these MOOAs are briefly introduced in the Appendix.

3.2. Parameter settings of the selected test algorithms

The performance of the optimization algorithms not only depends on the optimization problems but also on the settings of the algorithms, which affects solution quality and processing time. However, due to the lack of scientific information on parameter settings for building energy models, the default settings of the algorithm parameters were kept unchanged because they can be considered as general optimal settings for each
algorithm. Only the effects of changing the population size and the number of generations are investigated.

Population size is an important parameter affecting the performance of population-based optimization methods. In this study, the population sizes were twice / 4 times as many as the number of variables as recommended for population-based search methods [25] [26] (for small scale problem, the population size should be larger than the dimensionality of the problem). Thus two different choices were used in the performance tests as follows:
- Each generation has 64 individuals;
- Each generation has 32 individuals.

A detailed explanation of all parameters of these algorithms is beyond the scope of this paper. For further details of these parameter settings, we refer the reader to [27] for the NSGA-II and ENSES; to [28] for the MOPSO; to [19] for the PR_GA; to [29] for the evMOGA; to [30] for the spMODE-II; and to [31] for the MODA.

3.3. Test models

The studied house, Figure 2, is located in Helsinki, Finland, and has two floors with a net floor area of 143 m². The house is divided into 12 zones. Operable windows are installed in each zone for natural ventilation when the outdoor temperature is lower than the summer threshold 25 °C and higher than the heating set-point temperature of 21 °C. Wind pressure coefficients are used from Jokisalo (2009) to take account of the wind effect on infiltration and the natural ventilation of the house. No mechanical cooling is used. Internal shading control is assumed to apply the shading when the schedule is 'on' and the incident light exceeds 100 W/m² on the inside of the glass. The thermal performance of the house is evaluated by using the IDA-ICE 4.6 energy simulation program which was coupled with Matlab 2008a optimization toolbox to perform the optimization. The weather file used in the simulations was the reference year weather data (VantaaTRY2012) which was previously developed for energy simulation in Finland [32].

As a reference design, the house is assumed to be connected to the district heating grid with no on-site energy generation systems. The reference house is assumed to have minimum energy performance requirements in accordance with Finnish building regulations. The district heating plus the minimum energy saving measures (ESM) cost is about 21 k€. The reference design has an energy performance level of 150 kWh/m²a, and the total primary energy use that is a bit lower than the standard level 170 kWh/m²a, which came into force in the Finnish code [33].
In order to find a cost-optimal energy performance level in line with the recast EBPD of 2010, 16 design and operational variables were included in the optimization, as reported in Table 1. They were categorized into 3 groups:

- Energy saving measures: measures that reduce the energy demand (from \( X_1 \) to \( X_6 \)),
- Renewable energy sources: technologies that use renewable sources of energy (from \( X_7 \) to \( X_{11} \)),
- Mechanical systems: can be used if the demand cannot be satisfied through the renewable systems (from \( X_{12} \) to \( X_{15} \)).

The installation and maintenance costs of the energy supply systems (solar thermal collector, photovoltaic, and primary heating system) as well as their lifespan and their related annual subscription/energy delivering fees are based on the Finnish market 2013/2014. Further detailed about the design variables can be retrieved from [34]. As a consequence the total number of all possible combinations of the design options is \( \sim 1.6 \times 10^{10} \).

<table>
<thead>
<tr>
<th>Energy saving measures</th>
<th>Decision variables</th>
<th>X</th>
<th>No. possible options</th>
<th>Costs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Package of building envelope (PBenv.)</td>
<td>X1</td>
<td>8</td>
<td>From EUR 8000 to EUR 17000 (^1)</td>
<td></td>
</tr>
<tr>
<td>Efficiency of lighting and appliances</td>
<td>X2</td>
<td>2</td>
<td>According to [35] [36] (^2)</td>
<td></td>
</tr>
<tr>
<td>Type of heat recovery unit (efficiency %)</td>
<td>X3</td>
<td>3</td>
<td>EUR 1500, EUR 2000, EUR 2500 (^3)</td>
<td></td>
</tr>
</tbody>
</table>

\(^1\) \(^2\) \(^3\)
Efficiency of auxiliary systems (fans and pumps)  

<table>
<thead>
<tr>
<th>Efficiency of auxiliary systems (fans and pumps)</th>
<th>X4</th>
<th>2</th>
<th>EUR 800, EUR 1500,</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size of buffer tank (V_{tank})</td>
<td>X5</td>
<td>4</td>
<td>370 * V_{tank} + 1720 [37]</td>
</tr>
<tr>
<td>Insulation level of the buffer tank (Thins)</td>
<td>X6</td>
<td>4</td>
<td>150 * A_{tank} *Thins [38]</td>
</tr>
</tbody>
</table>

Area of solar thermal collectors  

| Area of solar thermal collectors | X7  | 8 | 492 A_{stn} + 500; |
| Area of photovoltaic module (A_{pv}) | X8  | 11 | 3.1A_{pv}^2 + 202A_{pv} + 1983; |

Overall efficiency of the photovoltaic  

| Overall efficiency of the photovoltaic | X9  | 2 | 70% of the given price |
| Slope angle of photovoltaic module | X10 | 3 | - |
| Azimuth angle of photovoltaic module | X11 | 4 | - |

Type of primary heating unit  

| Type of primary heating unit | X12 | 5 | According: [www.gebwell.fi](http://www.gebwell.fi) and [39] |
| Size of the primary heating unit | X13 | 13 | - |
| Supply water temperature from the primary heating unit (T_s) | X14 | 3 | - |
| Operating hour start at | X15 | 5 | - |
| Operating hour stop at | X16 | 5 | - |

*1 The price is calculated based on price assumptions from our previous paper [40]. Prices are updated assuming a 1.7% inflation rate, which is the average inflation rate of the last ten years [41]. The lifespan of the building envelope with the exception of the window is assumed to be 60 years. For windows, a lifespan of 30 years is assumed.

*2 Lifetimes for lighting options are modeled based on cumulative hours of use. 1200 and 10 000 operating hours are assumed for the incandescent and fluorescent lighting, respectively.

*3 The heat recovery is replaced every 15 years.

3.4. Objectives to be minimized

Two objective functions were used in this study and they are described as follows:

Min \( f_1(\bar{x}), f_2(\bar{x}) \), \( \bar{x} = [x_1, x_2, \ldots, x_m] \)  

(1)

where

- \( f_1 \): primary energy consumption (PEC).
- \( f_2 \): life-cycle cost (LCC) of the design solution (€/m²).
- \( \bar{x} \): combination of the design-variables \( (x_1, x_2, \ldots, x_m) \).
- \( m \): number of the design variables.

The tradeoff between these two objective functions are described in Figure 3, which also shows the financial and environmental gaps among the present building energy requirements and the nZEB and cost optimal buildings.

The PEC is minimized as the first objective function. The PEC considers the energy use of the house, including energy for heating, cooling, ventilation, lighting, pumps and fans, other technical service systems, domestic hot water systems, cooking, appliances, lighting) and...
the energy-saving by the renewable energy system. The hourly photovoltaic electricity production and the DHW energy saving were simulated by IDA ESBO program and checked by TRNSYS program.

The LCC is the sum of the present value of investment and operating costs for building and service systems, including those related to maintenance and replacement, including taxes, over a specified calculation period. To calculate the LCC of the house, this study assumed a 1.7% inflation rate; a 3.2% nominal interest rate. The average energy price was taken from the energy company Helsingin Energia [42]. The electricity, district heating and gas energy prices are assumed to be 12.5, 5, and 3.2 €/kWh, respectively. In order to evaluate the influence of financial assumptions on the optimization results, two different scenarios are considered as follows:

- Escalation rate of energy price “e” is 2% and Feed-in-tariff (FIT) is zero (no tariff);
- Escalation rate of energy price “e” is 6% and Feed-in-tariff (FIT) is 1 (full tariff).

3.5. Criteria for performance comparison

The performance of an optimization algorithm can be evaluated from some aspects which may give a comprehensive view. In this study, the performance of these seven algorithms is assessed using the following criteria and their corresponding indicators:

- Execution time of the algorithms,
- Convergence to the benchmarking optimal set, indicated by the normalized generational distance, denoted by GDn,
- Normalized inversed generational distance (IGDn) is used to quantify both the convergence and the diversity,
- Diversity of solutions in the Pareto-optimal set, indicated by the normalized diversity metric, denoted by DMn,
- Number of solutions on the Pareto-optimal set, denoted by NoPsolution,
- Contribution of the algorithm’s solutions to the best Pareto front.

As being introduced for the first time, the original generational distance [44, 45], inversed generational distance [46] and diversity metric [45] were calculated on the obtained optimal solutions for the two objective functions X and Y. To avoid the biases to one objective function, this study proposes using the normalized generational distance, normalized inversed generational distance and normalized diversity metric by calculating them on the obtained optimal solutions using the normalized values $X_n$ and $Y_n$ of the two objective functions as described by the following equations:

$$\begin{align*}
X_n &= \frac{X}{\text{Max}(X) - \text{Min}(X)} \\
Y_n &= \frac{Y}{\text{Max}(Y) - \text{Min}(Y)}
\end{align*}$$

The normalized generational distance GDn (a variant of the original GD) is a reliable convergence metric which does not have a bias to one of the objective functions. It indicates the average Euclidean distance between the best Pareto front $P^*$ and the optimal solution set $S$ obtained by each algorithm as follows:

$$GDn = \frac{1}{n} \left( \sum_{i=1}^{n} d_i^2 \right)^{1/2}$$

where $d_i$ is the Euclidean distance (in the objective space) between the obtained solution $i \in S$ and the nearest member of $P^*$; $n$ is the number of solutions in a Pareto front (see Figure 4). By definition, lower value of GDn is preferred.

The normalized inversed generational distance IGDn (a variant of the original IGD) is able to measure both the closeness and the diversity of the obtained solutions. The IGDn can measure the average distance between each member of $P^*$ and the obtained non-dominated solutions, it is therefore strongly influenced by the number of obtained non-dominated solutions and their distribution:

$$IGDn = \frac{1}{m} \left( \sum_{i=1}^{m} d_i^2 \right)^{1/2}$$

where $m$ is the number of solutions consisting in the best Pareto front; $d_i$ is the Euclidean distance between each member of $P^*$ and the nearest obtained non-dominated solution (see Figure 4). By definition, if the number obtained non-dominated solutions is small, the value of IGDn will grow. Once again, a lower value of IGDn is preferred.
To accurately calculate the $GD_n$ and $IGD_n$, a Pareto front with dense solutions is required. Thus in this study, the best Pareto front was generated by running the optimization 20 times on each algorithm, using two different algorithm’s settings (32 generations and 64 individuals/64 generations and 32 individuals). Best solutions from all these optimization runs were derived and they built up the benchmarking Pareto front. For each benchmarking Pareto front, 286720 model evaluations of 140 optimization runs were finished. As an example, Figure 5 shows the benchmarking Pareto front and all candidates of all obtained Pareto fronts found by the seven algorithms after 140 optimization runs.

*Figure 4: The concept of the $GD_n$, $IGD_n$, and $DM_n$ (adapted from [45, 46])*
Although the GDn and IGDn can give some information about the spread of the obtained solutions, the normalized diversity metric DMn can be used to accurately measure the diversity of the obtained solutions returned by an algorithm. The DMn was calculated as follows:

\[
DMn = \frac{d_f + d_l + \sum_{i=1}^{N} |d_i - \bar{d}|}{d_f + d_l + (N-1)\bar{d}}
\]  

(6)

where \(d_i\) is the Euclidean distance between the consecutive solutions in the obtained non-dominated set of solutions and \(\bar{d}\) is the average of all distances \(d_i\) \((i=1,..,N)\), assuming there are \(N\) solutions in the obtained non-dominated set. The parameters \(d_f\) and \(d_l\) are the Euclidean distances between the extreme and the boundary solutions (see Figure 4). This metric gives smaller values to better distributions and the most widely and uniformly spreadout set of non-dominated solutions returns a DMn of zero [45].

The number of the obtained non-dominated solutions of an algorithm is denoted by NoPsolution. Obviously, a high value of NoPsolution is preferred in multi-objective optimization.

Figure 5: The benchmarking Pareto (black squares) as well as all candidates of all obtained Paratos (140 Pareto fronts = 20 runs * 7 algorithms) using 200 (red) and 1800 (green) evaluations, respectively.
4. Performance of the optimization algorithms

As stated above, this study tests the performance of the 7 algorithms using two different population sizes (32 and 64 individuals) and two financial scenarios (FIT = 0, \( e = 2\% \) and FIT = 1, \( e = 6\% \)) to simulate the life-cycle cost. Consequently, there were four test cases as summarized in Table 2.

Table 2: The four test cases and their financial scenarios

<table>
<thead>
<tr>
<th>Test case</th>
<th>Population size (number of individuals in a generation)</th>
<th>Feed-in-tariff (FIT)</th>
<th>Escalation rate of energy price “( e )”</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>32</td>
<td>0</td>
<td>2%</td>
</tr>
<tr>
<td>A2</td>
<td>64</td>
<td>0</td>
<td>2%</td>
</tr>
<tr>
<td>B1</td>
<td>32</td>
<td>1</td>
<td>6%</td>
</tr>
<tr>
<td>B2</td>
<td>64</td>
<td>1</td>
<td>6%</td>
</tr>
</tbody>
</table>

In each test case and for each algorithm, optimization runs were done several times and consequently increasing number of generations. By this way, this study could examine the stability of the algorithms when the number of generations is increased and the convergence of the performance indicators. Figure 6 and Figure 7 show the optimization results of the two optimization algorithms – the PR_GA and the ENSES obtained after 3 and 27 generations, respectively, along with their performance indicators (the \( IGDn \) and \( DMn \)). It can be seen that the quality of the obtained solutions was improved when the number of generations was increased. Since it is hard to evaluate and compare the quality of the optimization solutions by using the graph only, these figures was accompanied by the \( IGDn \) and \( DMn \), which can be directly compared among the obtained solutions.

Figure 6: The best and the worst obtained tradeoff solutions of the PR_GA and the ENSES, in terms of \( IGDn \) and \( DMn \), compared with the benchmarking Pareto (Case A2 - after 3 generations)
4.1. Convergence to the best Pareto front and diversity of obtained solutions

Figure 8, Figure 9, Figure 10, and Figure 11 present the performance indicators of the 7 tested algorithms, resulting from 4 test cases A1, A2, B1, and B2 respectively. In these figures, for each indicator, each algorithm has 5 consecutive boxplots, corresponding to 5 different settings of the number of generations. Each boxplot shows the variation of the indicator’s sample generated by running the optimization 20 times.
In the test case A1, Figure 8 indicates that the PR_GA has the best performance indicators, followed by the spMODE-II which has consistent small $IGD_n$, $GD_n$ and $DM_n$. The PR_GA was therefore the strongest optimization method among these algorithms. The ENSES shows the worst convergence indicators, but it could give a good spreadout set of non-dominated solutions. In the test case A2, Figure 9 shows a similar result where the PR_GA outperformed the remaining algorithms while the ENSES returned poor converged solutions.

![Figure 8](image1.png)

Figure 8: The $IGD_n$, $GD_n$ and $DM_n$ of the tested algorithms - Test case A1: population size = 32 individuals; financial assumption: $FIT = 0$, $e = 2\%$. For each algorithm, optimization was executed 20 times, then the number of generations jumped from one to another value, i.e. from 6 to 18, 20, 42, 54 generations ($\approx 200, 600, 1000, 1400, 1800$ evaluations), respectively.

![Figure 9](image2.png)

Figure 9: The $IGD_n$, $GD_n$ and $DM_n$ of the tested algorithms - Test case A2: population size = 64 individuals; financial assumption: $FIT = 0$, $e = 2\%$. For each algorithm, optimization was executed 20 times, then the number of generations jumped from one to another value, i.e. from 3 to 9, 15, 21, 27 generations ($\approx 200, 600, 1000, 1400, 1800$ evaluations), respectively.

Figure 10 and Figure 11 demonstrate the results of the test case B1 and B2. Similarly, the PR_GA and the spMODE-II continue to show their superiority in terms of convergence and diversity. The ENSES always show the poorest convergence, but
acceptable diversity. The remaining algorithms – the NSGA-II, MOPSO, evMOGA and MODA were not competitive.

In conclusion, in terms of convergence and spread (given by the IGDn and GDn), the PR_GA was the best, followed by the spMODE-II. The pNSGA-II and the evMOGA exhibited average performance while the ENSES was almost the worst. In terms of diversity (given by the DMn), the PR_GA stood out as being the most efficient algorithms while differences among the rest were minor.

![Figure 10](image_url)

*Figure 10: The IGDn, GDn and DMn of the tested algorithms - Test case B1: population size = 32 individuals; financial assumption: FIT = 1, e =6%. For each algorithm, optimization was executed 20 times, then the number of generations jumped from one to another value, i.e. from 6 to 18, 20, 42, 54 generations (≈ 200, 600, 1000, 1400, 1800 evaluations), respectively.*
Based on the behavior of the boxplots (the average values of the distributions with a progressively higher number of generations) in Figure 8 - Figure 11, we could estimate the convergence rates of the algorithms to their Pareto optimal solutions. It can be observed that the pNSGA-II, MOPSO, evMOGA, ENSES and MODA reached their optimal solutions at a low number of evaluations, e.g. after about 1000 evaluations. Their solutions after 1000 evaluations showed minor improvements. Conversely, the PR_GA and the spMODE-II continued to improve their solutions considerably, even after 1800 evaluations. Their obtained solutions at 1800 evaluations differed significantly from those at 200 and 600 evaluations. Theoretically, when the number of generations is increased, the quality of the obtained solutions will be improved. In practice, this rule is not always true because of the stochastic nature of these algorithms (as shown in Figure 8 – Figure 11). In a few cases, the observed IGDn, GDn and DMn fluctuated, regardless of the number of generations. It is therefore important to compare performance of the optimization algorithms when convergence has been reached.

Figure 11: The IGDn, GDn and DMn of the tested algorithms - Test case B2: population size = 64 individuals; financial assumption: FIT = 1, e = 6%. For each algorithm, optimization was executed 20 times, then the number of generations jumped from one to another value, i.e. from 3 to 9, 15, 21, 27 generations (≈ 200, 600, 1000, 1400, 1800 evaluations), respectively.
4.2. Execution time

In this study, execution time of an algorithm was defined as average time needed to prepare all individuals for one generation. Figure 12 compares the execution time of 7 investigated algorithms. It can be seen the execution time of all algorithms varied considerably from the case A1 to case B2. In the test A1 and A2, the MOPSO, ENSES and MODA were the slowest algorithms while the remaining algorithms required short execution time. The situation changed significantly in the test B1 and B2 where only the slowest algorithm was recognized. In brief, it is hard to draw any solid results from the comparison of execution time.

4.3. Number of solutions on the Pareto-optimal set

Figure 13 shows the number of solutions on the Pareto-optimal set of the 7 algorithms. In all test cases, the pNSGA-II yielded the best results, followed by the PR_GA. The ENSES provided a limited number of non-dominated solutions, which can be used to explain the uncompetitive \( IGDn \) and \( GDn \) found in Figure 8 - Figure 11. In our study we observed that for a given population size, an increased number of generations (i.e. increased number of evaluations) resulted in more non-dominated solutions. As observed from Figure 13, the increased trends were almost leveled out at the 4\(^{th}\) and 5\(^{th}\) boxplots. It means that
about 1400 to 1800 evaluations were sufficient to stabilize the number of non-dominated solutions. More evaluations may not improve this value.

Although the NoPsolution can reveal the efficiency of the algorithms in finding non-dominated solutions, it cannot tell us the quality of these obtained solutions. This study therefore examined the number of solutions contributed by an algorithm to the best Pareto solutions of each optimization series. Each optimization series includes 20 optimization runs on each algorithm, and done on all 7 tested algorithms in turn. Table 3 shows the percentage of the best Pareto solutions contributed by each algorithm.

It is worthy of note that these contributions changed significantly when the total number of evaluations was increased. At convergence; i.e. optimization results after more than 1400 evaluations; the PR_GA contributes the most to the best Pareto solutions, from 50% in the test case A1 up to 73.8% in the test case B2. The second largest contributors are likely the evMOGA and spMODE-II. The remaining algorithms almost have no or negligible contribution, although at the earlier stage of the optimization (low number of evaluations) they tend to contribute the larger portions.

Figure 13: The NoPsolution of the 7 investigated algorithms in the four test cases (top to bottom: cases A1, A2, B1, and B2 respectively)
### Table 3: Contribution of each algorithm to the best Pareto solutions (%)

<table>
<thead>
<tr>
<th>Test case</th>
<th>(population size = 32, FIT = 0, e = 2%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number of evaluations</td>
</tr>
<tr>
<td>A1</td>
<td>200 boxplot 1</td>
</tr>
<tr>
<td></td>
<td>600 boxplot 2</td>
</tr>
<tr>
<td></td>
<td>1000 boxplot 3</td>
</tr>
<tr>
<td></td>
<td>1400 boxplot 4</td>
</tr>
<tr>
<td></td>
<td>1800 boxplot 5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test case</th>
<th>(population size = 64, FIT = 0, e = 2%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>B1</td>
<td>Number of evaluations</td>
</tr>
<tr>
<td></td>
<td>200 boxplot 1</td>
</tr>
<tr>
<td></td>
<td>600 boxplot 2</td>
</tr>
<tr>
<td></td>
<td>1000 boxplot 3</td>
</tr>
<tr>
<td></td>
<td>1400 boxplot 4</td>
</tr>
<tr>
<td></td>
<td>1800 boxplot 5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test case</th>
<th>(population size = 32, FIT = 1, e = 6%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number of evaluations</td>
</tr>
<tr>
<td>B1</td>
<td>200 boxplot 1</td>
</tr>
<tr>
<td></td>
<td>600 boxplot 2</td>
</tr>
<tr>
<td></td>
<td>1000 boxplot 3</td>
</tr>
<tr>
<td></td>
<td>1400 boxplot 4</td>
</tr>
<tr>
<td></td>
<td>1800 boxplot 5</td>
</tr>
</tbody>
</table>

### 4.4. Performance ranking and some observations

As this study includes many test cases, performance indicators on 7 tested algorithms, there were too many things to compare or/and examine. We therefore ranked the performance of the algorithms based on the indicators and then summarized these rankings in Table 4. By this way, we could obtain some clear observations as follows:

- In terms of convergence of the obtained solutions (indicated by the $IGD_n$ and $GD_n$), the PR_GA was superior to the others while the ENSES was the worst;
- In terms of diversity of obtained solutions (indicated by the $DM_n$), the PR_GA, ENSES and spMODE-II were slightly better than the remaining algorithms;
- In terms of execution time, none of the tested algorithms showed their outstanding performance, but it is easy to observe that the MOPSO and the ENSES were the slowest;
- In terms of the number of Pareto optimal solutions (indicated by the NoPsolution), the pNSGA-II and the PR_GA offered much more solutions than any other algorithms. The ENSES was the worst.
- Finally, in terms of contributions of the best solutions to the best Pareto fronts, once again the PR_GA outperformed the others while the MOPSO, the ENSES and the MODA were not competitive.

Table 4: Performance ranking of the 7 investigated algorithms, all performance criteria

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Test case</th>
<th>Ranking based on the performance indicators</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>IGDn</td>
</tr>
<tr>
<td>pNSGA-II</td>
<td>A1</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>A2</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>B1</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>B2</td>
<td>3</td>
</tr>
<tr>
<td>MOPSO</td>
<td>A1</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>A2</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>B1</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>B2</td>
<td>6</td>
</tr>
<tr>
<td>PR_GA</td>
<td>A1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>A2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>B1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>B2</td>
<td>1</td>
</tr>
<tr>
<td>evMOGA</td>
<td>A1</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>A2</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>B1</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>B2</td>
<td>4</td>
</tr>
<tr>
<td>ENSES</td>
<td>A1</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>A2</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>B1</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>B2</td>
<td>7</td>
</tr>
<tr>
<td>spMODE-II</td>
<td>A1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>A2</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>B1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>B2</td>
<td>2</td>
</tr>
<tr>
<td>MODA</td>
<td>A1</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>A2</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>B1</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>B2</td>
<td>5</td>
</tr>
</tbody>
</table>

*Algorithms that have the same rankings

In general, the PR_GA outperformed the others in most categories, including the most important ones. Conversely, the ENSES was the least convincing algorithm as it achieved low scores in most categories. In our test, the MODA and MOPSO did not exhibit any outstanding feature. The pNSGA-II, evMOGA and the spMODE-II achieved average
rankings in most tests thus they can be seen as alternative choices if the PR_GA is not readily available.

5. Discussion and conclusion

A comparison of the performance of seven multi-objective optimization algorithms, including the pNSGA-II, MOPSO, PR_GA, evMOGA, ENSES, spMODE-II and MODA has been carried out by means of computational experiments using a complex building energy model. We applied a multi-criteria system of performance indicators along with a comprehensive test procedure which allowed us to quantify many features of these MOOAs.

Two important assumptions of this study were that (i) we used most of the default settings of the algorithms (as recommended by the authors of the algorithms) and (ii) all algorithms were executed using the same number of evaluations/population sizes. These assumptions could provide a fair competition arena among the algorithms, but they may fail to enable the maximum power of these algorithms. As an example, if we test convergence of two algorithms with different convergence speeds using the same number of evaluations, the algorithm with the fast convergence speed will perform better. However, if we set another termination criterion (e.g. objective function convergence: changes of the objective functions are smaller than a user-specified threshold [1]), the obtained result might change. Hence the results of this study need to be interpreted carefully, accompanied by a full description of the assumptions.

In our test, the PR_GA consistently shows competitive performance, in terms of convergence, spread, number of non-dominated solutions, and contribution to the benchmarking Pareto. The pNSGA-II, evMOGA and spMODE-II were fairly stable and returned acceptable performance indicators. The ENSES, MOPSO and MODA were not competitive in most performance criteria. It is worthy of note that an algorithm may achieve a high score on one performance criterion, but obtain low scores on one or some other criteria. This is understandable as there is no single best algorithm for all optimization problems (no free lunch theorem).

The PR_GA showed outstanding scores in the test; however it is not an optimization algorithm. The PR_GA is intrinsically an optimization method in which a preparation phase is conducted for seeding the initial population with viable solutions before running the optimization using the GA. It is a well-established technique for improving optimization efficiency of evolutionary algorithms [47] and showed competitive performance in BOPs [48, 19, 49, 50]. Thus the PR_GA can be considered the first choice for BOPs. If we only consider the ‘single’ algorithms, the pNSGA-II, evMOGA and spMODE-II were the potential choices among the tested algorithms.

Performance of an optimization algorithm is undoubtedly important, especially in some areas where stringent criteria on system reliability and safety are usually required, e.g. structural engineering or aerospace engineering. However, selected optimization algorithms
should be made based on the purpose of the analysis, i.e. conceivably even the worst investigated MOOA could be good enough for the problem under investigated, or – alternatively – even the best one couldn’t be good enough. For example, in building design during the conceptual phase, a preliminary optimization run with a simple algorithm is sometimes sufficient for the task. Thus users remain the most important decision makers in selecting a suitable MOOA.

One important finding of this study is that none of the tested algorithms could converge completely to the best Pareto front within 1800 evaluations. The population size and the number of generations need to be defined carefully as the optimization result is sensitive to these values. This is because the stochastic nature of the population-based optimization algorithms which cannot assure a global minimum within a finite number of evaluations. We found that quality of the obtained solutions was gradually improved when the number of evaluations/generations increased. In our test, the performance indicators were likely to stabilize at 1400 to 1800 evaluations. We therefore recommend these values as minimum required number of evaluations for multi-objective optimization of building energy models. To maximize the power of a MOOA, we strongly recommend that researchers perform convergence tests to define the required number of evaluations.

As the main aim of this study is to enrich understanding on the natures of multi-objective optimization rather than to recommend the best algorithm for other applications, we try to provide an overall view of the performance and behavior of these algorithms based on which researchers can make a choice for their specific problem. In addition, performance of the MOOAs may be sensitive to test problems, which was not investigated in this study. Due to computational expensive simulations, the present tests were performed on one selected building energy model, thus we still have a question whether the building models have any influence on the test results. Knowledge and understanding about the MOOAs is therefore important in order to make a right choice.

6. Appendix

A brief introduction to the selected algorithms for the test

6.1. Non-dominated Sorting Genetic Algorithm II with a passive archive (pNSGA-II)

The NSGA-II is one of the most commonly-used multi-objective optimization algorithms, first proposed by Deb et al. [45]. The NSGA-II implements elitism by maintaining two populations of size N. The adult population P from the previous generation and the child population Q generated at the current generation. At each generation, these populations are combined and sorted according to the non-domination concept. Then N solutions are selected as the next parent population P. The number of non-dominated points available after sorting may be greater than the populations size N, which defines the number of (elite) points that are kept by the algorithm. When the number of the available non-
dominated points is greater than \( N \), the NSGA-II selects the \( N \) least crowded solutions by using the crowding distance measure and rejects the rest of the non-dominated points.

The pNSGA-II is the NSGA-II with a passive archive. The solutions that would be rejected by the original NSGA-II are saved in an archive. The passive archive works simply as storage for the evaluated solutions. Its members do not act in the solution generation procedure. As a last step of the algorithm, the non-dominated solutions are identified from the archive and are added to the final population.

The pNSGA-II with such kind of archive strategy was applied to a number of buildings and HVAC design optimization problems [3] and preliminary tests show its improved performance, compared with the NSGA-II [51].

6.2. Multi-objective particle swarm optimization

Eberhart and Kennedy [52] originally proposed the particle swarm algorithm for optimization. It is a swarm-based search algorithm which is relied on the simulation of the social behavior of bird flocking or fish schooling. In the particle swarm optimization (PSO) algorithm, potential solutions, called particles, move through the solution space by following the current optimal particles. The PSO has been found to be very effective in a wide variety of applications, being able to produce good optimal solutions at an acceptable computational cost [53]. The original PSO was only applicable to single-objective continuous optimization problems in which the decision variables are real numbers. Several modified PSO algorithms have been introduced using different approaches which make PSO capable in solving multi-objective discrete optimization problems [54]. In this study, the multi-objective PSO (MOPSO) proposed by Heris [28] was used. The algorithm of MOPSO can be retrieved from [55].

6.3. Two-phase optimization using Genetic algorithm (PR_GA)

The PR_GA was first proposed by Hamdy et al. [19]. PR_GA is not an optimization algorithm, but it is a combination of a preparation phase and main optimization one done by the genetic algorithm. The preparation phase improves the efficiency of optimization process by prepare a good initial population for the genetic algorithm.

By default, the multi-objective GA generates the initial population randomly using a creation function. The population of the next generation is prepared by comparing fitness of candidate solutions of the current generation using their non-dominated ranks and their distance measure. By this way, the next generation will track relatively the first generation. Random creation for the initial population and the dependence on this random behavior usually need a large number of trials in order to achieve good results [19].

By setting a good initial population, the best fitness at each generation and little diversity for the algorithm can easily be achieved. In addition, it allows the GA to focus on specified areas of the search space, which is close to the optimal Pareto front. For these reasons, a preliminary optimization run is done by deterministic algorithms in order to
prepare a good initial population for the GA before running the multi-objective genetic algorithm. The deterministic algorithms for the preparation process of the PR_GA were Fmincon solver and Fminimax function from Matlab 2008a optimization toolbox. In this study, a fast stopping criterion for these algorithms was implemented since the quality of the results is not the goal of this step.

6.4. Multi-objective evolutionary algorithm based on the concept of epsilon dominance (evMOGA)

The evMOGA is an elitist multi-objective genetic algorithm based on the concept of $\varepsilon$-dominance which is used to control the content of the archive where the solutions are stored. The evMOGA was first introduced in 2009 by Martinez-Iranzo et al. [29]. Different from the original MOGA, the evMOGA tries to search for a good approximation to the Pareto front with smart distributions (i.e. the density of solutions belonging to the Pareto front depends on its rate of change). By this way, the evMOGA is able to characterize the Pareto front with a small number of solutions, thereby reducing computational cost. The evMOGA also extends the limits of the Pareto front dynamically. Details on the $\varepsilon$-dominance and the evMOGA can be found in [56, 29].

6.5. Elitist non-dominated sorting evolution strategy (ENSES)

The ENSES algorithm is a variance algorithm based on the NSGA-II of Deb et al. [45]. The difference between the ENSES and the NSGA-II is that the ENSES uses Evolution Strategies (ES) instead of the genetic algorithm (GA) as the evolutionary algorithm for multi-objective optimization. Similar to the GA, evolution strategy also uses mutation, recombination and selection applied to a population of individuals containing candidate solutions in order to evolve iteratively into better and better solutions. The ENSES is implemented with an effective sorting method based on individual ranking, including a non-dominated sorting and a crowded distance metric sorting, which is similar to the sorting method used in the NSGA-II. It also shows good performance in providing widely distributed Pareto optimal solutions in multi-objective optimization [57].

6.6. Multi-objective differential evolution algorithm with spherical pruning based on preferences – an improved version of the spMODE algorithm (spMODE-II)

The DE is a very simple population-based, stochastic optimization algorithm, developed in 1994 - 1996 by Storn and Price [58]. The DE is a parallel direct search method which utilizes NP D-dimensional parameter vectors as a population for each generation G. The main concept of the DE is the scheme for generating trial parameter vectors, which is crucial for generating vectors of the next generation. Basically, the DE adds the weighted difference between two population vectors to a third vector. By this way no separate
probability distribution has to be used which makes the scheme completely self-organizing [58].

The multi-objective differential evolution (MODE) was first introduced in [59]. This MOOA was presented as a variant of the original DE, in which the best individual is adopted to generate the offspring. A Pareto-based approach is introduced to perform the selection of the best individual. Similar to the NSGA-II, MODE also uses ($\mu$+$\lambda$) selection, Pareto ranking and crowding distance in order to produce and maintain well distributed solutions. In [30], a spherical pruning with preferences MODE (called the spMODE-II) was presented and its code was kept open on Matlab Central (http://www.mathworks.com/matlabcentral), from which spMODE-II was implemented into this study.

6.7. MODA: Multi-Objective Dragonfly Algorithm

Mirjalili [31] proposed a new swarm-based algorithm - the dragonfly algorithm (DA) - that is capable in solving single-objective, discrete, and multi-objective problems. The major mechanism of the DA relies on the static and dynamic swarming behavior of dragonflies in nature, which is similar to the two main phases of optimization using meta-heuristics: exploration and exploitation. In the DA, these two phases are implemented by mimicking the social interaction of dragonflies in navigating, searching for foods, and avoiding enemies when swarming dynamically or statistically. To solve discrete problems, the author implemented the binary version of the DA, called binary DA. A multi objective dragonfly algorithm (MODA) was also introduced for use in multi-objective optimization. All details and source codes of the MODA can be found in [31] and on Mathworks.

References


