Chen, Yi and Wang, Zhonglai and Yang, Erfu and Li, Yun (2016) Pareto-optimality solution recommendation using a multi-objective artificial wolf-pack algorithm. In: The 10th International Conference on Software, Knowledge, Information Management and Application (SKIMA 2016), 2016-12-15 - 2016-12-17, Sichuan Province. ,
http://dx.doi.org/10.1109/SKIMA.2016.7916207

This version is available at https://strathprints.strath.ac.uk/59503/

Strathprints is designed to allow users to access the research output of the University of Strathclyde. Unless otherwise explicitly stated on the manuscript, Copyright © and Moral Rights for the papers on this site are retained by the individual authors and/or other copyright owners. Please check the manuscript for details of any other licences that may have been applied. You may not engage in further distribution of the material for any profitmaking activities or any commercial gain. You may freely distribute both the url (https://strathprints.strath.ac.uk/) and the content of this paper for research or private study, educational, or not-for-profit purposes without prior permission or charge.

Any correspondence concerning this service should be sent to the Strathprints administrator: strathprints@strath.ac.uk
Pareto-Optimality Solution Recommendation Using A Multi-Objective Artificial Wolf-Pack Algorithm

Yi Chen, Zhonglai Wang, Erfu Yang, Yun Li

Abstract—In practical applications, multi-objective optimisation is one of the most challenging problems that engineers face. For this, Pareto-optimality is the most widely adopted concept, which is a set of optimal trade-offs between conflicting objectives without committing to a recommendation for decision-making. In this paper, a fast approach to Pareto-optimal solution recommendation is developed. It recommends an optimal ranking of decision-makers using a Pareto reliability index. Further, it provides users with a recommendation list of optimal ranking of optimal diversities for decision-making. In this paper, we propose a fast approach of Pareto-optimal solution recommendation using a non-dominated sorting method (MAWNS). This is tested in a case study, where the MAWNS is employed as an optimiser for a widely adopted standard test problem, ZDT6. The results show that the proposed method works valuably for the multi-objective optimisations.

I. INTRODUCTION

Real-world applications, such as structural optimisations of space systems[1], parameters determination for financial market quantitative modelling, Terahertz spectroscopic analysis for drug or explosive mixture[2], and intelligent analysis for ‘Big Data’ management, often imply multiple objectives. Thus, people need to search for ‘trade-offs’, rather than a single solution, which leads to the different solution of ‘optimality’ under the multi-objective situations. The most widely used term is the notion of Pareto optimality [3], [4], [5].

A multi-objective optimisation problem could be written in Equ. (1),

\[
\text{maximise : } F_i(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_N(\mathbf{x})], \quad i = 1, 2, \cdots, N
\]

Subject to the equality constraints \( G_i(\mathbf{x}) \), as given in equation (2),

\[
G_i(\mathbf{x}) = 0, \quad i = 1, 2, \cdots, M
\]

and the inequality constraints \( H_i(\mathbf{x}) \), as given in equation (3),

\[
H_i(\mathbf{x}) \leq 0, \quad i = 1, 2, \cdots, P
\]

where, \( N \) is the number of objective functions, the objective functions \( F_i : \mathbb{R}^n \rightarrow \mathbb{R} \); \( M \) is the number of the equality constraints; \( P \) is the number of the inequality constraints; \( \mathbf{x} = [x_1, x_2, \cdots, x_K] \) is the decision variables vector, \( K \) is the number of the variables.

All vectors satisfied equations (2) and (3) are named as the set \( \mathcal{Y} \), in which the particular set of \( \mathbf{x}^* = [x_1^*, x_2^*, \cdots, x_K^*] \) yields the optimum values of all the objectives. The vector of decision variables \( \mathbf{x}^* \in \mathcal{Y} \) is Pareto optimal if there is no feasible vector of decision variables \( \mathbf{x} \in \mathcal{Y} \) which would increase some criterion without causing a simultaneous decrease in any other criterion. The vectors \( \mathbf{x}^* \) corresponding to the solutions included in the Pareto optimal set are called non-dominated. The image of the Pareto optimal set under the objective functions is called Pareto front[3], [4], [5].

There are some widely used algorithms to solve multi-objective formulations, such as the non-dominated sorting genetic algorithm (NSGA)[6], the niched-Pareto genetic algorithm (NPGA)[7], the multi-objective genetic algorithm (MOGA)[8], the strength Pareto evolutionary algorithm 2 (NSGA-II)[12], [13], the niched Pareto genetic algorithm 2 (NPGA2)[14], the Pareto archived evolution strategy (PAES)[11], the nondominated sorting genetic algorithm II (NSGA-II)[12], [13], the niched Pareto genetic algorithm 2 (NPGA2)[14], the Pareto envelope-based selection algorithm (PESA-I)[15], the revised version of Pareto envelope-based selection algorithm (PESA-II)[16], the micro-genetic algorithm (μGA)[17], [18], the micro-genetic algorithm with variable population size (VP_{μGA}) [19] and the bat algorithm for multi-objective optimisation [20], [21], etc.

Generally, an engineer can make trade-offs within this set under practical requirements by focusing the set of Pareto front choices, which provides a visualised demonstration of the Pareto-optimal solution, but with an unclear indication of optimal diversities for decision-making. In this paper, we propose a fast approach of Pareto-optimal solution recommendation (FPR) using the Pareto reliability index (PRI), which provides users with a recommendation list of optimal ranking and optimal trend indications with different risk tolerances.

Wolves are always regarded as one of the smartest animals...
on Earth, and wolves are gregarious animals who mostly live in packs - wolf-pack (WP). Inspired by the swarm intelligence of the WP’s dynamic behaviours, a multi-objective artificial wolf-pack algorithm (MAWPA) has been developed, which can respond quickly to the environmental changes and their neighbours in the direction and speed, etc., the information of their behaviours can be transferred to others and help them move from one swarming configuration to another almost as one unit. By borrowing this intelligence of the social behaviours, the MAWPA is parallel and independent to the initial values, and able to achieve a global optimum. In this paper, the MAWPA is proposed to handle multi-objective optimisations, which is employed for a standard test problem ZDT6.

The remainder of this paper is organised as follow. Section I introduces the background of this research work; Section II defines the technical work-flow of MAWPA; Section III describes the FPR technical roadmap; Section IV states the normalisation for the fitness functions; Section V defines the PRI factor; Section VI introduces two trend indices for the evolutionary process; Section VII gives two case studies to demonstrate the FPR method; Section VIII concludes this paper.

II. MULTI-OBJECTIVE ARTIFICIAL WOLF-PACK ALGORITHM

Inspired by the swarm intelligence of the WP’s dynamic behaviours, the MAWPA is an artificial intelligent algorithm that firstly simulates the behaviour of an individual artificial Wolf (AW) and constructs a WP. Each AW searches its own local optimal solution and passes information to its self-organised WP, and finally, achieves the global optimal solution. The MAWPA work-flow is given in Fig. 1, which includes 6 steps of operations: (1) initialisation; (2) behaviour selection; (3) behaviour of scouting; (4) behaviour of calling; (5) behaviour of besieging; (6) bulletin and (7) non-dominated sorting using the non-dominated sorting genetic algorithm II (NSGA-II)[13].

Initialisation: in this step, all the parameters will be initialised, and the programme is preparing itself for the next steps.

Behaviour Selection: the behaviour selection step takes ‘Scouting’ as the default behaviour or initial behaviour for each WP. According to the density of prey in this region, the number of companion and the visual conditions.

Scouting: for a certain AW individual $k$, $S_k = \{s_1, \ldots, s_M\}$ is its finite state set, there is $M$ states that an AW can perform in. Within the AW’s visual field, if the current state of this AW is $S_i$ and the next state is $S_j$, the AW moves from $S_i$ to $S_j$ randomly and check the state updating conditions as stated in Equations (4) and (5). As demonstrated in Fig. 2, $r_{ij} = \|S_j - S_i\|$ is the distance between the $i^{th}$ and $j^{th}$ individual AW. $F = f(S)$ is the prey density for this AW, where $f$ is the fitness function. $\delta$ is the iterate step, $\epsilon$ is the random moving factor. $\nu$ is the AW visual constant.

\[
S_{t+1} = \begin{cases} 
S_i + \epsilon \cdot \delta \cdot \frac{S_j - S_i}{\|S_j - S_i\|} & \text{if } F_j > F_i \\
S_i + \epsilon \cdot \delta & \text{otherwise}
\end{cases}
\] (4)

\[S_j = S_i + \epsilon \cdot \nu\] (5)

Calling: suppose the number of this AW’s neighbours is $\gamma$, the central state is $S_c$, the prey density is $F_c = f(S_c)$ and $\eta$ is the crowd factor. Within its visual field ($r_{ij} < \nu$), if the $F_c/\gamma > \eta F_i$ and $\eta \geq 1$, the AW implements the central state driven step; otherwise, when the $F_c/\gamma \leq \eta F_i$ or $\eta = 1$, the AW will go on with the scouting behaviour, as expressed in Equation (6).
\[ S_{i+1} = \begin{cases} S_i + \epsilon \cdot \delta \cdot \frac{S_c - S_i}{\|S_c - S_i\|} & \text{if } \frac{F_c}{\gamma} > \eta F_i \text{ and } \eta \geq 1 \\ \frac{F_c}{\gamma} \leq \eta F_i & \text{or } \eta = 0 \end{cases} \] (4)

**Besieging:** when the AW’s companions reach “max” state \( S_{\text{max}} \) with the number \( \gamma \) within the neighbourhood, the prey density reaches \( f_{\text{max}} \) at the mean time. As stated in Equation (7), with the same conditions as Equation (6), the AW updates its state in highest prey density region; otherwise, the AW will go on with the searching behaviour, as expressed in Equation (6).

\[ S_{i+1} = \begin{cases} S_i + \epsilon \cdot \delta \cdot \frac{S_{\text{max}} - S_i}{\|S_{\text{max}} - S_i\|} & \text{if } \frac{F_{\text{max}}}{\gamma} > \eta F_i \text{ and } \eta \geq 1 \\ \frac{F_{\text{max}}}{\gamma} \leq \eta F_i & \text{or } \eta = 0 \end{cases} \] (7)

**Bulletin:** the bulletin operation is a step to compare each AW’s current state \( S_i \) with the historical state data, the bulletin data will be replaced and updated only when the current state is better than the last one, as described by Equation (8).

\[ S_{j+1} = \begin{cases} S_j & \text{if } F_j > F_i \\ S_i & \text{otherwise} \end{cases} \] (8)

A “max-generation” of simulation is employed as the terminal condition of the MAWPA programme, which is one of the widely used criteria for optimisation.

**III. FAST APPROACH OF PARETO-OPTIMAL SOLUTION RECOMMENDATION**

As shown in Fig. 3, the flow chart of the fast approach of Pareto-optimal solution recommendation is divided into 6 steps:

- step 1, initialise parameters and start the optimisation process;
- step 2, perform optimisation using multi-objective algorithms;
- step 3, Pareto-optimal solutions generation;
- step 4, the PRI assessment block;
- step 5, check optimisation termination conditions;
- step 6, end the programme and post-calculation process.

As can be seen from Fig. 3, there are 4 sub-steps in the FPR assessment block in step 4,

- sub-step 1, normalisation for the multi-objective fitness functions of the Pareto-optimal solutions;
- sub-step 2, the PRI index \( \beta_1 \) calculation;
- sub-step 3, calculation of the evolutionary trend indices, the mean average precision(mAP) and the mean standard deviation(mSTD);
- sub-step 4, visualisation of Pareto front and evolutionary trend indices;

![Flowchart](image)

**IV. NORMALISATION**

The normalisation process is to map variables from their original value range to a normalised value range, e.g. \([0, 1]\), by two operations of **scale** and **shift**\([23]\). As defined in equation (1), the vector of objective functions \( f_i \) with the values \([f_1(x), f_2(x), \ldots, f_N(x)]\) is the original data source to the normalisation block, in which \( f_i \in [f_{\text{min}}, f_{\text{max}}] \).

Firstly, as given in equation (9), the **scale** operation calculates the scale factor according to the input range \([f_{\text{min}}, f_{\text{max}}]\) of the original data \( f_i \), and then all the input data are scaled to the range of \([c_1, c_u]\). That is, the fitness values are mapped from the practical value range \([f_{\text{min}}, f_{\text{max}}]\) to the normalised value range \([c_1, c_u]\), which are \([0, 1]\) in this context.

\[ f_s = (c_u - c_1) \times \frac{f_i - f_{\text{min}}}{f_{\text{max}} - f_{\text{min}}} \] (9)

Then, in the **shift** operation, the scaled data \( f_s \) are shifted to the new range of \([c_1, c_u]\), as given in equation (10), where \( f_{\text{sh}} \) is the normalised fitness objectives.

\[ f_{\text{sh}} = c_1 + f_s \] (10)

**V. PARETO RELIABILITY INDEX FOR PARETO SOLUTIONS**

The Pareto reliability index \( \beta_1 \) is defined in equation (11), where \( \mu_f \) is the mean and \( \sigma_f \) is the standard deviation of the normalised objectives, as given in equations (12) and (13) respectively. \( W_i \) is the weighted normalised objectives in equation (14), \( w_i \) is the weight factor as given in equation (15),
which balances the weight of all the normalised objectives, \( w_i \in [0, 1] \).

\[
\beta_1 = \frac{\mu_f}{\sigma_f} 
\]  
(11)

\[
\mu_f = \frac{\sum_i^N W_i}{N} 
\]  
(12)

\[
\sigma_f^2 = \frac{\sum_i^N (W_i - \mu_f)^2}{N - 1} 
\]  
(13)

\[
W_i = w_i \cdot f_{ih} 
\]  
(14)

\[
\sum_i^N w_i = 1 
\]  
(15)

Fig. 4. Pareto Reliability Index[22]

As shown in Fig. 4, without loss of generality, a case of two objectives \( f_1 \) and \( f_2 \) is utilised to present the definition of \( \beta_1 \). Fig. 4 shows a geometrical illustration of the \( \beta_1 \) index in a dual-objective case, which indicates the distance of the mean margin of a multi-criteria range. The idea behind the \( \beta_1 \) is that the distance from location measure \( \mu_f \) to the limit states \( \sigma_f \) which provides a good measure of the reliability of the Pareto solutions, that is, a larger value of \( \beta_1 \) leads to a better solution.

VI. TRENDS INDICES

In this section, a factor of mAP and a factor of mSTD are introduced as the trend indices for the optimisation process, which are defined in equations (16) and (17).

As shown in Fig. 5, the solid curve is the mAP scores for each vector \( f_j \) as given in equation (16) and the dashed curves are the mAP ± mSTD for each vector \( f_j \) as given in equation (17), in which \( p \) is the population of the data set, AVG(\( \cdot \)) is the average function and VAR(\( \cdot \)) is the variance function.

\[
\text{mAP}(f_j) = \frac{1}{p} \sum_{j=1}^{p} (\text{AVG}(f_j)) 
\]  
(16)

\[
\text{mSTD}(f_j) = \frac{1}{p} \sum_{j=1}^{p} \left( \sqrt{\text{VAR}(f_j)} \right) 
\]  
(17)

Fig. 5. The diagram of mAP ± mSTD over the full generations[22]

VII. CASE STUDY

As given by equation (18), a standard test problem ZDT6 [24] is solved by NSGA-II implemented in the MATLAB toolboxes SGALAB [19], [25], Swarmwolf[26] and SECFLAB[27], in which \( x_i \in [0, 1] \), \( n = 10 \) in this context.

\[
\begin{align*}
    f_1(x) &= 1 - \exp \left( -4x_1 \right) \sin \left( 6\pi x_1 \right) \\
    f_2(x) &= g(x) \left[ 1 - \left( \frac{f_1(x)}{g(x)} \right)^2 \right] \\
    g(x) &= 1 + 9 \left[ \frac{\sum_{i=2}^{n} x_i}{n-1} \right]^{0.25}
\end{align*}
\]  
(18)

The parameters for the case are listed in Table I, in which a max-generation 200 is the termination condition of each round test; the total test number is 10; the population is 30, tournament selection operator, binary encoding/decoding method, single point crossover and mutation operators with \( p_c = 0.8 \) and \( p_m = 0.01 \) respectively.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>max-generation</td>
<td>200</td>
</tr>
<tr>
<td>crossover probability ( (p_c) )</td>
<td>0.8</td>
</tr>
<tr>
<td>mutation probability ( (p_m) )</td>
<td>0.01</td>
</tr>
<tr>
<td>population</td>
<td>60</td>
</tr>
<tr>
<td>selection operator</td>
<td>tournament</td>
</tr>
<tr>
<td>crossover operator</td>
<td>single point</td>
</tr>
<tr>
<td>mutation operator</td>
<td>single point</td>
</tr>
<tr>
<td>encoding/decoding method</td>
<td>binary</td>
</tr>
<tr>
<td>( P_N ) non-replaceable population</td>
<td>50</td>
</tr>
<tr>
<td>( P_R ) replaceable population</td>
<td>10</td>
</tr>
<tr>
<td>( \delta ) iterate step</td>
<td>0.5</td>
</tr>
<tr>
<td>( v ) visual</td>
<td>2.5</td>
</tr>
<tr>
<td>( \eta ) crowd</td>
<td>0.618</td>
</tr>
<tr>
<td>try number</td>
<td>5</td>
</tr>
</tbody>
</table>

Figs. 6 and 7 are the mAP ± mSTD diagrams for \( f_1 \) and \( f_2 \) over the full simulation generations, which indicate that a
better solution of $f_1$ can be optimised without worsening a solution of $f_2$, which is not dominated by any other solution in the search space. As can be seen in Fig. 6, the $f_1$’s mAP ± mSTD curves go up quickly from generation = 1 to 8, when they reach a stable status with a minor fluctuation at generation = 10 and last to the end of simulation. Fig. 7 shows the $f_2$’s mAP ± mSTD curves have a similar shape of generation = 1 to 8, and they have a second jump from generation = 100 to 130 and then keep stale to the end of simulation.

As shown in Table II, the recommended solutions are listed by the solution number in the column ‘SOLUTION No.’ with a descending rank (RANK = 1 is the most recommended) using the values of $\beta_1$.

<table>
<thead>
<tr>
<th>RANK</th>
<th>SOLUTION No.</th>
<th>$\beta_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>41</td>
<td>2.25e15</td>
</tr>
<tr>
<td>2</td>
<td>43</td>
<td>2.25e15</td>
</tr>
<tr>
<td>3</td>
<td>43</td>
<td>2.25e15</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>21</td>
<td>218</td>
<td>6.83</td>
</tr>
<tr>
<td>22</td>
<td>219</td>
<td>6.83</td>
</tr>
<tr>
<td>23</td>
<td>220</td>
<td>6.83</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>44</td>
<td>15</td>
<td>4.63</td>
</tr>
<tr>
<td>45</td>
<td>16</td>
<td>4.63</td>
</tr>
<tr>
<td>46</td>
<td>17</td>
<td>4.63</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>300</td>
<td>270</td>
<td>0.499</td>
</tr>
</tbody>
</table>

Fig. 6. $f_1$’s mAP ± mSTD over the full generations

Fig. 7. $f_2$’s mAP ± mSTD over the full generations

VIII. Conclusions and Future Works

Using the newly defined index of $\beta_1$, a fast approach to Pareto-optimal solutions recommendation has been developed, thereby providing a ranking list of Pareto-optimal solutions for the decision-making. The evolutionary trends are gauged via the indices of mAP ± mSTD with variable uncertainty tolerances.

The contributions of this paper includes: (1) the inclusion of the dynamic behaviours of trend indices of mAP and mSTD; (2) the development of a fast Pareto-optimality solution recommendation method, FPR; (3) the Pareto reliability index $\beta_1$ or PRI to rank the uncertainties of Pareto-optimal solutions, and a clear recommendation list for decision-making; (4) a multi-objective artificial wolf-pack swarm algorithm using non-dominated sorting method, MAWNS.

Further work aims at industrial applications, including the multi-objective optimisations for robotic systems, such as exoskeleton, robotic space tethers, humanoid robot and industrial robotics. Further, it will be applied to decision processes for computational intelligence aided design [28][29].

REFERENCES


