Abstract - In this paper, we test the convergence speed of the fast elitist Non-Dominated Sorting Genetic Algorithm (NSGA-II) on the Gross Domestic Product (GDP) quarterly disaggregation problem. In fact, we perform many simulations by considering various inputs of the NSGA-II parameters with respect to the population size (pop size), the iteration number (no_rum), the maximum number of generations (gen_max) and the mutation distribution index in the polynomial mutation (etam). It turns out that for suitably large values of the parameters pop size, no_rum and gen_max, the NSGA-II algorithm converges to a single Pareto-optimal solution located in a fixed cuboid. Due to the elitism in the NSGA-II and our simulation results, we suspect that the mutation distribution index alone doesn’t influence the time of algorithm convergence and the number of Pareto-optimal solutions. Therefore, we reach the most likely conclusion that only an appropriate choice of the parameter triplet (pop size, no_rum, gen_max) can ensure the convergence of the algorithm toward a fixed Pareto-optimal solution for the Quadratic Multi-objective Programming of GDP quarterly disaggregation. It is worth noted that our work is an extension of previous works in the same field.

Keywords - Simulation, Convergence, Multiobjective Optimization, NSGA-II Algorithm, Pareto-optimal front, Quarterly Gross Domestic Product

I. INTRODUCTION

Several algorithms have been developed in the literature to solve multi-objective optimization problems, but most of these multi-objective optimization problems are Non Polynomial (NP)-difficult according to their size. Generally, all the algorithms can be classified into two types of approaches: (i) non-Pareto approaches that seek to reduce the initial problem to single-objective problems and (ii) Pareto approaches that do not transform the objectives of the problem and treat them simultaneously by seeking better compromise solutions [2] according to the resolution process.

A multi-objective programming problem is generally formulated as presented in equation (1):

\[
\min_{x \in \mathbb{R}^N} F(x) = (f_1(x), f_2(x), \ldots, f_Q(x))
\]  

where \( F: \mathbb{R}^N \rightarrow \mathbb{R}^Q \) is the vector of objective functions, \( f_i \) represents the \( i \)-th objective function, \( Q \) is the number of objective functions and generally \( Q \geq 2 \) and \( U \subseteq \mathbb{R}^N \) the set of all the equality and inequality constraints. A solution \( x \) is a vector of \( N \) decision variables: \( x = (x_1, x_2, \ldots, x_N), x \in \mathbb{R}^N \).

In the literature, many multi-objective evolutionary algorithms have been suggested for solving multi-objective programming. Recently, it has been proved that simulation results on difficult test problems showed that the Non-dominated Sorting Genetic Algorithm (NSGA)-II, in most problems, is able to find much better range of solutions and better convergence speed near the true Pareto-optimal front [4]. NSGA-II is a very effective algorithm but is often criticized for its computational complexity.

The selected population and the chosen parameters generate offsprings from crossover and mutation operators. So a bad choice of the parameters could affect negatively the convergence of the algorithm.

The step-by-step process shows that NSGA-II algorithm is simple and straightforward. NSGA-II is a better sorting algorithm which incorporates elitism and no sharing parameter needs to be chosen a priori. Since it is not possible to find a solution that simultaneously optimizes all objective functions in the case of a multi-objective program, the dominance in the sense of Pareto is introduced in the algorithm.

The fastness of NSGA-II and the convergence speed of the algorithm depend on the choice of the parameters. A test on the parameters will be discussed in detail in this paper.

In fact, it is difficult to ensure perfect cooperation between objective functions in the search for Pareto optimal solutions with a lot of objective functions. In fact, a multi-objective optimization program gives multiple Pareto optimal solutions in general.

For the sake for looking for a suitable algorithm that solving the quarterly disaggregation of Gross Domestic Product (GDP) as a multi-objective programming, our aim in this paper is to test the robustness of the convergence and the speed towards single Pareto-optimal solution, using the solving algorithm proposed in [8]. Therefore, we test how fast the elitist multi-objective NSGA-II can converge towards a fixed Pareto optimal point if we control the key parameters, mainly, the mutation distribution index in the polynomial mutation. For this purpose, we use the multi-
Objective programming problem of GDP quarterly disaggregation developed in [7]. This paper is an extension of previous works in the same field. In section II, we present the quarterly disaggregation problem of GDP. The structure of solving algorithm in described in section III and the simulation results a discussed in section IV.

II. THE QUARTERLY DISAGGREGATION PROBLEM OF GDP

The quarterly disaggregation of Gross Domestic Product (GDP) problem is the new one proposed and developed in [7]. All the elementary objective functions identified are grouped in equation (2) with the constraints given by equations (3)-(7) to form the above multi-objective programming:

$$\min_{x} \{ f_1(x), f_2(x), f_3(x) \} \tag{2}$$

Subject to

$$X = (X_1, X_2, X_3) \tag{3}$$

$$X_k = (X_k)_t \quad \forall \ t \quad k \tag{4}$$

$$-X_{k,t} \leq 0 \quad \forall \ t, k \tag{5}$$

$$\sum_{t=y-3}^{y} X_{k,t} - Z_{k,y} = 0 \quad \forall y, k \tag{6}$$

$$\sum_{t=y-3}^{y} X_{k,t} \eta_{k,t} = Z_{k,y} \sum_{t=y-3}^{y} \eta_{k,t} \quad \forall k, y \tag{7}$$

Where

- $T$: the number of years for national accounts observed,
- $y \in \{1, 2, 3, ..., T\}$, year generic index,
- $i \in \{1, 2, 3, 4\}$, quarterly index,
- $t \in \{1, 2, 3, ..., 4T\}$: generic index of quarters on the $T$ years’ period obtained from $i$ and $y$ using an operator proposed in [6];
- $k \in \{1, 2, 3\}$, generic branch index of the quarterly accounts nomenclature;
- Value of the annual account per branch: let $Z_{k,y}$ be the (known) value added of the branch account $k$, for the year $y$;
- quarterly indicator: let $I_{k,t}$ be the value of the branch indicator $k$, for the quarter $t = 1, 2, 3, ..., 4T$;
- $I_k = (I_{k,t})_{t=1,2,4T}$: the vector of quarterly indicator related the branch $k$ over the entire period;
- Inter-branch interaction: we note $W_{j,k}$ the interaction of the branch $j$ on the branch $k$, considered as the average share of the branch $k$ demand of product coming from the branch $j$.

$W_{j,k} = 1$ if $j = k$ and $0 \leq W_{j,k} < 1$ if $j \neq k$.

- Value added per branch: let $X_{k,t}$ be the value of national account for branch $k$ at quarter $t = 1, 2, 3, ..., 4T$;
- The vector of national account value for branch $k$ over the entire period is noted: $X_k = (X_{k,t})_{t=1,2,4T} \quad X_k \in \mathbb{R}^{4T} \quad for \quad k = 1, 2, 3$;
- The vector of quarterly national accounts value for all branches over the entire period is noted: $X = (X_1, X_2, X_3), X \in \mathbb{R}^{4T \times 3}$;
- Thus, the objective functions are given by equation (8):

$$f_k(X_1, X_2, X_3) = \sum_{j=1}^{3} \sum_{t=2}^{4T} W_{j,k} \left( Y_{j,t} - \frac{Y_{j,t-1}}{Y_{j,t-1}} \right)^2 \tag{8}$$

III. THE ALGORITHM DESIGN

The algorithm for GDP quarterly disaggregation problem presented as a multi-objective programming in [8], [9], has the architecture as presented in Figure 1.

![Figure 1. Architecture of the GDP quarterly disaggregation algorithm](image)

The core of the model simulation is based on the NSGA-II algorithm developed in [1], [4], [10] which is adapted to the problem of GDP quarterly disaggregation in [8]. The algorithm is a fast elitist multi-objective programming algorithm. We provide the bounded values (minimum and maximum) of the target variables in the NSGA-II algorithm. Finally, the pseudo code of the complete algorithm is given by the above steps:

**Step 1:** Setting the decision space upper bound and lower bound for the target variables.

**Step 2:** Initializing all general parameters

**Step 3:** Interpolation of accounts with quarterly indicators

**Step 4:** Initialization of the other NSGA-II input parameters number of constraints, number of variables, the size of initial population to be generated, the number of iteration, the maximum number of generations
We use in this paper a computationally fast and elitist multi-objective evolutionary algorithm based on NSAG-II. On the test problem taken from [7], NSGA-II is able to converge closer to a single fixed Pareto-optimal front with the GDP quarterly disaggregation problem. However, NSGA-II maintains diversity among solutions by controlling the mutation distribution index in the polynomial mutation.

In the convergence speed test by controlling both the mutation distribution index and the parameter triplet \((\text{pop}_\text{size}, \text{no}_\text{run}, \text{gen}_\text{max})\), it is clear that the mutation distribution index alone doesn’t influence the speed of algorithm convergence. However, we note that such a deterministic crowding coupled with the effect of mutation-based approach has been beneficial for convergence close to a fixed Pareto-optimal solution.

Finally, our conclusion is that, if the parameter triplet \((\text{pop}_\text{size}, \text{no}_\text{run}, \text{gen}_\text{max})\) is judiciously chosen so that \(\text{pop}_\text{size} \geq 100, \text{no}_\text{run} > 1\) and \(\text{gen}_\text{max} > 2\), the fast and elitist NSGA-II algorithm converges to a fixed Pareto optimal solution located in a fixed cuboid for the test problem for any value of the mutation distribution index.

### TABLE 1: SENSIBILITY OF MUTATION DISTRIBUTION INDEX ON THE CONVERGENCE OF THE GDP PROGRAM ALGORITHM

<table>
<thead>
<tr>
<th>((\text{pop}<em>\text{size}, \text{no}</em>\text{run}, \text{gen}_\text{max}))</th>
<th>Mutation distribution index (etam)</th>
<th>Running Time (in second)</th>
<th>Number of Pareto-optimal solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>((10, 1, 25))</td>
<td>0</td>
<td>18.292</td>
<td>10</td>
</tr>
<tr>
<td>((10, 1, 25))</td>
<td>1</td>
<td>19.2736</td>
<td>8</td>
</tr>
<tr>
<td>((10, 1, 25))</td>
<td>10</td>
<td>19.6863</td>
<td>9</td>
</tr>
<tr>
<td>((10, 1, 25))</td>
<td>100</td>
<td>17.6145</td>
<td>9</td>
</tr>
<tr>
<td>((10, 1, 25))</td>
<td>500</td>
<td>18.4421</td>
<td>9</td>
</tr>
<tr>
<td>((10, 1, 25))</td>
<td>1000</td>
<td>19.0442</td>
<td>8</td>
</tr>
<tr>
<td>((1, 1, 25))</td>
<td>2000</td>
<td>17.6917</td>
<td>6</td>
</tr>
<tr>
<td>((1, 1, 25))</td>
<td>5000</td>
<td>18.6984</td>
<td>8</td>
</tr>
<tr>
<td>((100, 10, 25))</td>
<td>0</td>
<td>1656.67</td>
<td>1</td>
</tr>
<tr>
<td>((100, 10, 25))</td>
<td>100</td>
<td>1703.82</td>
<td>1</td>
</tr>
<tr>
<td>((100, 10, 25))</td>
<td>200</td>
<td>1733.95</td>
<td>1</td>
</tr>
<tr>
<td>((100, 10, 25))</td>
<td>500</td>
<td>1769.96</td>
<td>1</td>
</tr>
<tr>
<td>((100, 10, 25))</td>
<td>1000</td>
<td>1751.14</td>
<td>1</td>
</tr>
</tbody>
</table>
Figure 2. Optimal Pareto solutions for different simulation parameters $(pop\_size, no\_runs, gen\_max)$ with etam=100
Figure 3. Various Optimal Pareto solutions for different simulation parameter etam values with afixed (pop_size, no_run, gen_max)
ACKNOWLEDGMENT

All simulations are performed with Octave software installed on HP Intel Core i7 (vPro) notebook PCs, 16 Gb RAM, under Windows system.

We thank all the reviewers for their comment. We would like to send special thanks to the African Centre of Excellence in Mathematical Sciences and Application (ACE-MSA) at Institute of Mathematics and Physics Sciences (IMSP) for all its supports.

REFERENCES


