ABSTRACT
In this paper a comparison between the single and multi-objective based Optimization techniques including GA, PSO and Hybrid will be presented. The hybrid technique is combined of two attractive evolutionary techniques, Particle Swarm Optimizer (PSO) and Genetic Algorithm (GA) to enhance the search process by improving the diversity, and the convergence toward the preferred solution. This is not only modeled on the concepts of natural selection and evolution (GA) but also based on cultural and social rules derived from the analysis of the swarm intelligence and from the interaction among particles (PSO). This enables a faster convergence without degrading the estimated set of solutions. Indeed, the population diversity is correctly conserved during the optimization process. The previously experiment done by the researchers are retested to show the comparison among the optimization techniques.

Keywords
Multi-Objective Particle Swarm Optimization(MOPSO), Multi-Objective Genetic Algorithm(MOGA), Hybrid Optimization

1. INTRODUCTION
Several modern heuristic tools have evolved that facilitate solving optimization problems that were previously difficult or impossible to solve. These tools include evolutionary computation, simulated annealing, tabu search, particle swarm, etc. The genetic algorithm (GA) and particle swarm optimization (PSO) techniques appeared as promising algorithms for handling the optimization problems. These techniques are finding popularity within research community as design tools and problem solvers because of their versatility and ability to optimize in complex multimodal search spaces applied to non-differentiable cost functions. GA can be viewed as a general-purpose search method, an optimization method, or a learning mechanism. PSO is inspired by the ability of flocks of birds, schools of fish, and herds of animals to adapt to their environment, find rich sources of food, and avoid predators by implementing an information sharing approach[2].

Genetic Algorithms (GA) and Particles Swarm Optimization (PSO) are both population based algorithms that have proven to be successful in solving a variety of difficult problems. However, both models have strengths and weaknesses. Comparisons between GAs and PSOs have been performed by Eberhart and Angeline and both conclude that a hybrid of the standard GA and PSO models could lead to further advances [1]. In a multi-objective based GA/PSO, a wider range of alternatives is usually identified and the models will be more realistic by the multi-objective methodology.

We begin by reviewing the single and multi-objective PSO and GA in Section II, and then introducing the hybrid technique of PSO and GA in more detail in Section III. Section IV presents the performance comparison on the PSO, GA, and Hybrid model that was introduced in the previous work. Section V and VI presents the conclusion with suggestions for future work and acknowledgement.

2. OPTIMIZATION TECHNIQUES

2.1 Single Objective Optimization
The main goal of single objective (SO) optimization is to find the “best” solution, which corresponds to the minimum or maximum value of a single objective function that lumps all different objectives into one. This type of optimization is useful as a tool which should provide decision makers with insights into the nature of the problem [6]. The single objective based PSO and GA are described in the following sections.

2.1.1 Particle Swarm Optimization(PSO)
The Particle Swarm Optimization Algorithm (PSO) is a population-based optimization method finds the optimal solution using a population of particles(individual) [2,3]. Every swarm of PSO is a solution in the solution space. PSO is basically developed through simulation of bird flocking in two-dimension space.

The PSO heuristic was first proposed by Kennedy and Eberhart [15] for the optimization of continuous non-linear functions. A fixed population of solutions is used, where each solution (or particle) is represented by a point in N-dimensional space. The ith particle is commonly represented [2, 6, 24, 26] as
selected to reproduce. A GA technique is composed of five steps:

**Step 1:** Initialize the population randomly. The first step consists in randomly initializing a population of individuals. The size of the population and the coding of the individual are defined by the designer and can have some influence on the algorithm convergence (speed).

**Step 2:** Evaluate the population. A weight is given to each individual according to its fitness.

**Step 3:** Generate new individuals. Parents can be selected either randomly or according to their fitness.

**Step 4:** Evaluate the population. A GA will continually make its population evolve.

**Step 5:** Keep the population size constant.

An advantage of the GA techniques is that they lead, in most of the cases, to the global optimal Pareto frontier.

### 2.2 Multi-Objective Optimization

In a multi-objective optimization with conflicting objectives, there is no single optimal solution. The interaction among different objectives gives rise to a set of compromised solutions, largely known as the trade-off, non-dominated, non inferior or Pareto-optimal solutions. Multi-objective methodologies are more likely to identify a wider range of these alternatives since they do not need to pre-specify for which level of one objective a single optimal solution is obtained for another [6].

#### 2.2.1 Multi-Objective PSO (MOPSO)

In the Multi-objective PSO a local lbest is found for each swarm member selected from the 'closest' two swarm members instead of a single gbest. The concept of closeness is calculated in terms of only one of the evaluated objective dimensions, with the selection of the local optima from the two based upon the other objective [7].

The selection of which objective to fix (used to find the 'closest') and which to optimise is based on the knowledge of the test function design and the relatively simple objective function being fixed.

---

\[ X_i = (x_1,1; x_2,2; \ldots; x_i, N) \]

and its performance evaluated on the given problem and stored. Each particle maintains knowledge of its best previous evaluated position, represented as \( P_i = (p_{i,1}; p_{i,2}; \ldots; p_{i,N}) \), and also has knowledge of the single global best solution found so far, in the traditional single objective application indexed by \( g \). The rate of position change of a particle then depends upon its previous local best position and the global best, and its previous velocity. For particle \( i \) this velocity is \( V_i = (v_{i,1} \ldots v_{i,N}) \). The general algorithm for the adjustment of these velocities is:

\[
V_{i,j} = w v_{i,j} + c_1 r_1 (p_{i,j} - x_{i,j}) + c_2 r_2 (p_g, j - x_{i,j})
\]

\[
X_{i,j} = x_{i,j} + \chi v_{i,j}; j = 1; \ldots; N
\]

Where \( w \) is the inertia of a particle, \( c_1 \) and \( c_2 \) are constraints on the velocity toward global and local best, \( \chi \) is a constraint on the overall shift in position, \( r_1; r_2 \sim U(0; 1) \).

During each generation each particle is accelerated toward the particle’s previous best position and the global best position. At each iteration a new velocity value for each particle is calculated based on its current velocity, the distance from its previous best position, and the distance from the global best position. The new velocity value is then used to calculate the next position of the particle in the search space.

This process is then iterated a set number of times, or until a minimum error is achieved. In the inertia version of the algorithm an inertia weight, reduced linearly each generation, is multiplied by the current velocity and the other two components are weighted randomly to produce a new velocity value for this particle, this in turn affects the next position of the particle during the next generation.

#### 2.1.2 Genetic Algorithm (GA)

GAs are stochastic, non-linear optimization routines loosely based on theories of biological evolution, mechanics of natural selection and natural genetics. They are search or exploration algorithms generally used as optimization techniques to search the global optimum of a function. However, they can also advantageously be used in other fields: as example on applications where robustness and global optimization are needed [4].

GAs have been developed by John Holland at the University of Michigan. With the closely related evolutionary algorithms, GAs are a class of non-gradient methods: actually, in contrast to more traditional optimization methods, which use gradient information to move towards better points in solution space, GAs operate on populations of solutions using models of natural selection.

In GAs, each optimization parameter (\( x_i \)) is encoded by a gene using an appropriate representation, such as a real number or a string of bits. The corresponding genes for all parameters \( x_1; \ldots; x_n \) form a chromosome capable of describing an individual design solution. A set of chromosomes representing several individual design solutions compose a population where the fittest are selected to reproduce. A GA technique is composed of five steps:
This is shown in Figure 1 with the nearest particles to \( b \) highlighted (in terms of the 'simpler' objective 2), meaning the \( \text{best} \) for \( b \) is \( c \) (the fitter of the two neighbors in terms of objective 1). A single \( P_{\text{best}} \) is maintained for each swarm member, which is only replaced when a new solution is found which dominates it (identical to the 'conservative' preservation of efficiency selection rule.

This is demonstrated in Figure 1 with particle \( a \) moving to a fitter position at generation \( t + 1 \) (one that dominates its previous position). This new position is mutually non dominating with the \( p_{\text{best}} \) of \( a \), however, as the multi-objective evaluation of the new particle does not lie in the lower quadrant of the \( p_{\text{best}} \) (represented in Figure 1 with a square), \( Pa \) remains unchanged. The performance of the MOPSO was demonstrated on a number of test functions.

### 2.2.2 Multi-Objective GA

A multi-objective GA optimization can be defined as the problem of finding a set of design variables (DV) which optimizes a set of Objective Function (OF) and simultaneously satisfies a set of constraint functions. A multi-objective optimization problem can be expressed as follows:

Find DV set:

\[
X = (x_1, x_2, ..., x_n)^T \in (DVS)
\]

which minimizes OF:

\[
f(X) = (f_1(x), f_2(x), ..., f_k(x))^T \in (OFS)
\]

and simultaneously satisfies constraints:

\[
h_i(x) = 0, i \in [1, ..., q_h] \quad \text{(equality constraints)}
\]

\[
g_i(x) \leq 0, i \in [1, ..., q_g] \quad \text{(inequality constraints)}
\]

where:

- \( n \) is the number of DV (optimization parameters) which belong to the design variables space (DVS);
- \( k \) is the number of OF to be optimized; OF are included in the objective function space (OFS). The space of DV can contain both discrete and continuous variables;
- \( q_h \) is the number of equality constraint functions;
- \( q_g \) is the number of inequality constraint functions.

The flow chart of Multi-objective GA is as follows:

3. **HYBRID OPTIMIZATION**

A proper combination of the GA and PSO models could produce a very effective search strategy [3,4]. The hybrid algorithm combines the standard velocity and position update rules of PSOs with the ideas of selection, crossover and mutation from GAs. The basic PSO and GA hybrid framework is depicted below:

---

**Figure 2:** Hybrid Optimization Framework

**Figure 3:** Flow chart of the Hybrid Optimization
4. PERFORMANCE COMPARISON

Five numeric optimization problems were chosen to compare the relative performance of the Hybrid optimization algorithm to GA and PSO. These functions are standard bench-mark test functions and are all minimization problems[2].

4.1 Test Problems

Ellipsoidal function:
\[ f_1(x) = \sum_{i=1}^{n} i x_i^2 \]

Rosenbrock function:
\[ f_2(x) = \sum_{i=1}^{n} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2) \]

Rastrigin Function:
\[ f_3(x) = \sum_{i=1}^{n} (x_i^2 - 10 \cos(2\pi x_i) + 10) \]

Griewank function:
\[ f_4(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos \left( \frac{x_i}{\sqrt{i}} \right) + 1 \]

Ackley function:
\[ f_5(x) = 20 + e - 20e^{-0.2\sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2}} - e^{\frac{1}{n} \sum_{i=1}^{n} \cos(2\pi x_i)} \]

<table>
<thead>
<tr>
<th>Function</th>
<th>Symmetric Initialization Range</th>
<th>Asymmetric Initialization Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f_1 )</td>
<td>(-100, 100)</td>
<td>(50, 100)</td>
</tr>
<tr>
<td>( f_2 )</td>
<td>(-30, 30)</td>
<td>(15, 30)</td>
</tr>
<tr>
<td>( f_3 )</td>
<td>(-5.12, 5.12)</td>
<td>(2.56, 5.12)</td>
</tr>
<tr>
<td>( f_4 )</td>
<td>(-600, 600)</td>
<td>(300, 600)</td>
</tr>
<tr>
<td>( f_5 )</td>
<td>(-32.768, 32.768)</td>
<td>(16.384, 32.768)</td>
</tr>
</tbody>
</table>

4.2 Result

Table 2 and 3 show the mean best fitness value along with the standard deviation for each test case using symmetric initialization and asymmetric initialization, respectively. In many test cases the PSO algorithm was able to find near optimal solutions in a majority of trials. For the majority of these cases the population would converge on a sub optimal solution preventing the population from improving. In general the GA performed poorly when using asymmetric initialization. However, using symmetric initialization the GA was able to outperform at least one of the PSO versions in a number of cases, in particular function \( f_3 \) in all dimensions and function \( f_4 \) in all dimensions (see Table 2).

5. CONCLUSION & FUTURE WORK

In this paper the single and multi-objective GA, PSO and Hybrid optimization techniques have been discussed. It is indicated that most of the time the Hybrid optimization technique performs better than others. The future works include combining the MOGA and PSO to get better optimization result.

6. ACKNOWLEDGMENTS

A special thanks to Research Management Center (RMC) UTM that supported us to do this project.

7. REFERENCES


Table 2: Symmetric Initialization

<table>
<thead>
<tr>
<th>Problem</th>
<th>Dims</th>
<th>Gens</th>
<th>GA Mean Best (std-dev)</th>
<th>PSO mean best (std-dev)</th>
<th>Hybrid mean best (std-dev)</th>
</tr>
</thead>
<tbody>
<tr>
<td>f1</td>
<td>20</td>
<td>1500</td>
<td>0.0001172 (3.37E-05)</td>
<td>2.17E-37 (5.54E-39)</td>
<td>1.69E-40 (5.81E-40)</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>2000</td>
<td>0.000271 (7.01E-05)</td>
<td>3.59E-27 (1.30E-26)</td>
<td>2.99E-30 (8.65E-30)</td>
</tr>
<tr>
<td>f2</td>
<td>20</td>
<td>1500</td>
<td>25.86 (22.99)</td>
<td>30.36 (42.41)</td>
<td>7.412 (1.867)</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>2000</td>
<td>38.81 (23.74)</td>
<td>57.52 (47.46)</td>
<td>32.3 (24.1)</td>
</tr>
<tr>
<td>f3</td>
<td>20</td>
<td>1500</td>
<td>0.002937 (0.0008124)</td>
<td>10.51 (3.337)</td>
<td>0.4179 (0.60)</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>2000</td>
<td>0.004746 (0.001144)</td>
<td>24.74 (5.708)</td>
<td>1.91 (1.721)</td>
</tr>
<tr>
<td>f4</td>
<td>20</td>
<td>1500</td>
<td>0.01952 (0.0236)</td>
<td>0.026 (0.02631)</td>
<td>0.003152 (0.006287)</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>2000</td>
<td>0.003944 (0.005308)</td>
<td>0.01495 (0.01623)</td>
<td>0.003989 (0.006789)</td>
</tr>
<tr>
<td>f5</td>
<td>20</td>
<td>1500</td>
<td>0.003292 (0.000591)</td>
<td>5.67E-15 (1.60E-15)</td>
<td>3.25E-15 (9.64E-16)</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>2000</td>
<td>0.003587 (0.000474)</td>
<td>1.57E-14 (5.83E-15)</td>
<td>1.93E-14 (5.95E-14)</td>
</tr>
</tbody>
</table>
Table 3: Asymmetric Initialization

<table>
<thead>
<tr>
<th>Problem</th>
<th>Dimension</th>
<th>Generations</th>
<th>GA Mean Best (std-dev)</th>
<th>PSO mean best (std-dev)</th>
<th>Hybrid mean best (std-dev)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_1$</td>
<td>20</td>
<td>1500</td>
<td>5.328 (14.56)</td>
<td>22.75 (26.93)</td>
<td>19.31 (54.09)</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>2000</td>
<td>47.88 (28.60)</td>
<td>50.19 (48.57)</td>
<td>36.2 (34.42)</td>
</tr>
<tr>
<td>$f_2$</td>
<td>20</td>
<td>1500</td>
<td>22.75 (26.93)</td>
<td>17.36 (1.765)</td>
<td>6.419 (3.865)</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>2000</td>
<td>17.12 (1.647)</td>
<td>57.52 (47.46)</td>
<td>32.3 (24.1)</td>
</tr>
<tr>
<td>$f_3$</td>
<td>20</td>
<td>1500</td>
<td>1.457 (1.816)</td>
<td>10.51 (3.337)</td>
<td>0.4179 (0.60)</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>2000</td>
<td>10.13 (4.469)</td>
<td>24.74 (5.708)</td>
<td>1.91 (1.721)</td>
</tr>
<tr>
<td>$f_4$</td>
<td>20</td>
<td>1500</td>
<td>416.9 (11.17)</td>
<td>0.026 (0.02631)</td>
<td>0.00478 (0.00987)</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>2000</td>
<td>416.9 (0.04309)</td>
<td>0.01495 (0.01623)</td>
<td>2.0387 (1.32E-18)</td>
</tr>
<tr>
<td>$f_5$</td>
<td>20</td>
<td>1500</td>
<td>19.78 (0.0249)</td>
<td>2.009 (6.029)</td>
<td>3.25E-15 (9.64E-16)</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>2000</td>
<td>19.83 (0.218)</td>
<td>5.604 (8.987)</td>
<td>9.86E-15 (9.52E-15)</td>
</tr>
</tbody>
</table>