Multi-Objective Particle Swarm Optimization Algorithm (MOPSO)

Prof. DSc. BARSEKH-ONJI Aboud

Universidad Anáhuac México
Facultad de Ingeniería

November 19, 2025

Contents

Agenda

Why Multi-Objective Optimization?

Beyond a Single Goal

So far, our PSO algorithm has focused on finding **one** best solution (the gbest) for a **single** objective (e.g., 'minimize error' or 'find the minimum of a function').

Why Multi-Objective Optimization?

Beyond a Single Goal

So far, our PSO algorithm has focused on finding **one** best solution (the gbest) for a **single** objective (e.g., 'minimize error' or 'find the minimum of a function'). But what happens when we have more than one goal at the same time?

Real-World Problems are Conflicting

Most real-world problems have several (often conflicting) objectives.

- Minimize the **cost** of a bridge...
- ...while maximizing its **safety**...
- ...and minimizing its **weight**.

The Multi-Objective Optimization Problem (MOP)

Formal Definition

We want to find the vector of decision variables $\vec{x}^* = [x_1^*, ..., x_n^*]^T$ which will satisfy:

- m inequality constraints: $g_i(\vec{x}) \leq 0$
- p equality constraints: $h_i(\vec{x}) = 0$

and will **optimize** the vector function:

$$\vec{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), ..., f_k(\vec{x})]^T$$

Where k is the number of objectives (and for MOPs, $k \geq 2$).

Why Not Use Classical Methods?

Limitations of Classical Programming

Techniques like linear programming or gradient descent are often not ideal for MOPs.

- They tend to generate elements of the optimal set **one at a time**. You would have to run the algorithm many times.
- They are very sensitive to the **shape** of the final trade-off curve (the 'Pareto front'). They often fail if the front is concave or disconnected.

Why Use a Population-Based Metaheuristic?

The Power of a Population (a Swarm)

Metaheuristics (like the PSO) are ideal for MOPs because they deal simultaneously with a **set of possible solutions** (the population).

Key Advantages

- We can find **several members** of the optimal 'trade-off' set **in a single run**.
- They are **not sensitive** to the shape or continuity of the solution set (they can easily handle concave or disconnected fronts).

Why Use a Population-Based Metaheuristic?

Key Advantages

- We can find **several members** of the optimal 'trade-off' set **in a single run**.
- They are **not sensitive** to the shape or continuity of the solution set (they can easily handle concave or disconnected fronts).

We just need to adapt our PSO algorithm to handle multiple objectives instead of one. This is what **MOPSO** does.

Agenda

The 'Optimum' Changes

Rethinking 'Best Solution'

In single-objective PSO, our goal was simple: find the single best solution (gbest).

The 'Optimum' Changes

Rethinking 'Best Solution'

In single-objective PSO, our goal was simple: find the single best solution (`gbest`). In a Multi-Objective Problem (MOP), this is no longer true. There is no single 'best' solution.

We are looking for 'Trade-Offs'

Instead of a single solution, we are trying to find a set of good **compromises** (or **'trade-offs'**).

The 'Optimum' Changes

Rethinking 'Best Solution'

In single-objective PSO, our goal was simple: find the single best solution (`gbest`).
In a Multi-Objective Problem (MOP), this is no longer true. There is no single 'best' solution.

We are looking for 'Trade-Offs'

Instead of a single solution, we are trying to find a set of good **compromises** (or **'trade-offs'**).
For example, a solution that is *slightly* more expensive (worse) but *much* safer (better) is a valid compromise. We want to find all such compromises and let the human designer choose one.

A Brief History: Edgeworth and Pareto

Francis Ysidro Edgeworth (1881)

The notion of optimality we use was first proposed by Francis Ysidro Edgeworth.

Vilfredo Pareto (1896)

This notion was later generalized by the economist Vilfredo Pareto.
We will use the most commonly accepted term: **Pareto Optimum**.

Pareto Optimality: Formal Definition

Definition

A vector of decision variables $\vec{x}^* \in \mathcal{F}$ is **Pareto Optimal** if there does not exist another vector $\vec{x} \in \mathcal{F}$ such that:

$f_i(\vec{x}) \leq f_i(\vec{x}^*)$ for all $i = 1, ..., k$

and

$f_j(\vec{x}) < f_j(\vec{x}^*)$ for at least one objective j.

Pareto Optimality: In Simple Words

Explanation

In plain English, a solution $\vec{x}^*$ is **Pareto Optimal** if there is no other solution $\vec{x}$ that can improve at least one objective without making any other objective worse.

- **Solution A:** Cost=100, Safety=9
- **Solution B:** Cost=90, Safety=9
- **Solution C:** Cost=90, Safety=8

- **B dominates A**: B is cheaper (better) and has the same safety.
- **B dominates C**: B is just as cheap and has better safety.
- Solution **B** is **nondominated** by A or C.

The Key Vocabulary

The Goal of a MOPSO

- **Nondominated Solution:** A solution that is not dominated by any other feasible solution in the search space. (Like Solution B in the previous example).
- **Pareto Optimal Set:** The set of **all** nondominated solutions.
- **Pareto Front:** The plot of the objective function values (e.g., the Cost vs. Safety values) for all solutions in the Pareto Optimal Set.

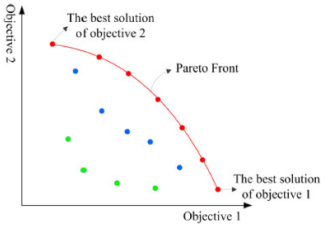

Figure 1: The **Pareto Front** is the set of optimal trade-offs.

Agenda

Revisiting Single-Objective PSO

The Algorithm We Know

In the standard PSO (gbest model) that we already studied, every particle updates its velocity based on two simple, clear pieces of information:

- Its own best-known position (`pbest`).
- The **single best-known position** found by the entire swarm (`gbest`).

The 'gbest' Velocity Update

The key was this equation, which guided the entire swarm toward one single point:

$$v_{ij}(t+1) = wv_{ij}(t) + c_1 r_1(y_{ij}(t) - x_{ij}(t)) + c_2 r_2(\hat{y}_j(t) - x_{ij}(t))$$

Where $\hat{y}_j(t)$ was the `gbest`.

Why Does Standard PSO Fail for MOPs?

The Problem with gbest and pbest

The entire logic of PSO is built on the idea that there is **one** best solution to find. In a Multi-Objective Problem, we don't have one gbest. We have an entire **Pareto Front** of solutions that are all 'equally optimal' (nondominated).

The Algorithm Breaks

We can no longer answer these simple questions:

- If gbest is now a *set* of 100 solutions, which one should the particle follow?
- If a particle finds a new solution, how does it know if it's 'better' than its current pbest? It might be better in one objective but worse in another.

Why Does Standard PSO Fail for MOPs?

Examples

Conclusion To create a MOPSO, we must fundamentally change how PSO's core components work.

The 3 Fundamental Challenges

1. Leader Selection

How do we select a particle (leader) from the Pareto Front to guide the swarm?

- If all nondominated solutions are equally good, which one should a particle use as its `gbest`?
- Should the choice be random, or based on another criterion?

The 3 Fundamental Challenges

2. Solution Retention

How do we retain the many nondominated solutions found during the search?

- We can't just store one `pbest` and one `gbest` anymore.
- We need a special place to store all the optimal trade-offs we find.

The 3 Fundamental Challenges

3. Diversity Maintenance

How do we maintain diversity to prevent the entire swarm from converging to a single point on the Pareto Front?

- It's not enough to find *one* optimal solution. We must find the *entire* front.
- We need a mechanism to spread the particles out.

Agenda

Solving Retention: The External Archive

Solution for Challenges 1 & 2

To solve the problems of **leader selection** and **solution retention**, most MOPSOs use an **External Archive** (also called a repository).

What is an External Archive?

It is a special, separate list that stores all the nondominated solutions found so far during the entire search. It becomes the 'memory' of the swarm.

Solving Retention: The External Archive

Update Rules for the Archive

- When a particle flies to a new position, the new solution is compared to the archive.
- The new solution is added to the archive **only if** it is not dominated by any solution already in the archive.
- If the new solution **dominates** one or more solutions in the archive, those dominated solutions are removed.

The Bounded Archive Problem

A New Problem: Infinite Growth

The external archive has a practical drawback: its size can grow very quickly. Updating and managing an archive with thousands of solutions at every generation becomes computationally expensive.

The Solution: A Bounded (Fixed-Size) Archive

Due to practical reasons, archives are given a maximum size (e.g., 100 solutions).
But this creates a new question: If the archive is full and a new, valid nondominated solution is found, which solution do we remove from the archive?

The Bounded Archive Problem

Maintaining Diversity in the Archive

We need an additional criterion. The most common solution is to remove the solution that is in the **most crowded region** of the Pareto front. This helps to keep the solutions in the archive **well-spread**.

Solving Leader Selection

Picking a Leader from the Archive

Now that the swarm has an archive of nondominated solutions, how does a particle choose its `gbest` (its leader) from this set?

Using Density to Select a Leader

A good strategy is to select leaders from the **less crowded** regions of the archive. This encourages particles to fly towards the 'empty' parts of the Pareto front, helping to spread out the solutions.

Solving Leader Selection

Nearest Neighbor Estimator

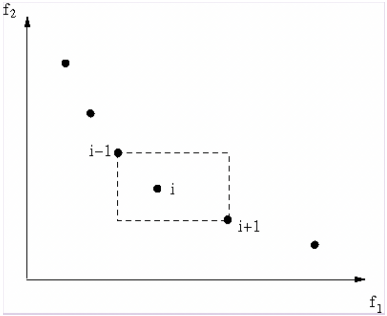

Figure 2: Prefers solutions with a larger 'cuboid' (less crowded).

Kernel (Niching) Estimator

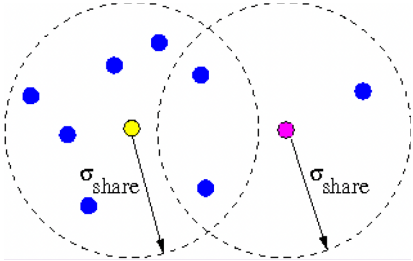

Figure 3: Prefers solutions with fewer neighbors within a 'niche' radius σ_{share}.

Solving Diversity: Flight Strategy (Mechanism 1)

Besides selecting leaders, we can promote diversity in two main ways: (1) the flight strategy and (2) mutation.

Mechanism 1: Flight Strategy

- **Topologies:** Using neighborhoods smaller than the entire swarm (like in `lbest` PSO) can preserve diversity for longer.
- **Inertia Weight (w):** The inertia weight (w) is crucial for controlling the trade-off between global (wide) and local (nearby) exploration.
- A common strategy (Shi, 1999) is to **linearly decrease** w during the run. This allows for more global exploration at the beginning and more fine-grained local search at the end.

Solving Diversity: Mutation (Mechanism 2)

Mechanism 2: Mutation (Turbulence)

The standard PSO velocity update is like a 'mutation with a conscience,' as it is guided by the particle's own experience and its neighbors' experience. Sometimes, however, we need 'craziness' or 'turbulence'—a random mutation.

Why is Mutation Needed?

- A swarm can **stagnate** when particle velocities become almost zero.
- This leads to the whole swarm being trapped in a **local optimum**.
- A mutation operator introduces a random change that can 'kick' a particle out of the local optimum and potentially find a new, better region of the search space.

The General MOPSO Algorithm

1. **Initialize** the swarm (positions $P[i]$ and velocities $V[i]$).
2. **Initialize Archive:** Evaluate the swarm and store all nondominated particles in the external archive.
3. **Loop** until stopping condition is met:
   - **Select Leaders:** For each particle $P[i]$, select a leader from the archive (e.g., using a density measure).
   - **Fly:** Update the particle's velocity and position using `pbest` and selected leader.
   - **Mutate:** Apply a turbulence/mutation operator to the particle's position.
   - **Evaluate:** Calculate the objective values for the particle's new position.
   - **Update `pbest`:** Update $P[i]$'s `pbest` if the new position is not dominated by its old `pbest`.
   - **Update Archive:** Add the new particle to the archive if it is nondominated. Remove any solutions from the archive that are now dominated by this new particle.
4. **End Loop**.

Agenda

Where is MOPSO Used?

MOPSOs are used to solve complex, real-world problems:

- Optimal groundwater management
- Design of efficient speed profiles for Automatic Train Operation (ATO)
- Planning of electrical distribution systems, incorporating distributed generation
- Complex network clustering
- Partial classification for accident severity analysis
- And many more in engineering, data mining, and scheduling

What is Missing? The Future of the Field

MOPSO is a very active field. Key research areas are still open:

Impact of Parameters

- Detailed studies on the impact of parameters are still missing.
- The role of leader selection was underestimated for many years.
- MOPSOs that can adapt their own parameters are still very scarce.

Theoretical Studies

- Critical aspects like **convergence analysis** and **run-time analysis** for MOPSOs are currently missing in the literature.

What is Missing? The Future of the Field

New Algorithms

- More creativity is needed in designing new algorithms.
- Use for **Many-Objective Optimization** (problems with 4 or more objectives) is a new and growing area.

Applicability

- We need to identify *which types* of problems MOPSOs are best suited for.
- Many authors use MOPSO for its simplicity, but cannot show a clear advantage over other evolutionary algorithms for their specific problem.

What is Missing?

Incorporation of Preferences

- The work on incorporating a user's preferences into a MOPSO is scarce.
- This is very important for real-world problems, as a designer may not want to see thousands of solutions on a Pareto front.

Parallelism

- Few parallel MOPSOs exist.
- Implementations on **GPUs** are also scarce.
- Implementations in hardware do not seem to be available.

Agenda

Conclusions

What's Next?

As the research area matures, it is desirable to start analyzing deeper questions:

- **Applicability:** What problems are MOPSOs really good (or bad) at solving?
- **Search Behavior:** How does a MOPSO actually move through the search space compared to a genetic algorithm?
- **Theoretical Aspects:** We need to develop a solid theoretical foundation for convergence and run-time.

Final Thought

This work is expected to appear within the next few years as this field reaches its maturity.