1. BINARY PSO (BPSO)

- Core Concept: Inspired by bird flocking/fish schooling

- Key Parameters:

  * w (inertia weight): Controls velocity inheritance (0.7-0.9)

  * c1 (cognitive): Individual learning rate (typically 2.0)

  * c2 (social): Social/group learning rate (typically 2.0)

  * Population size: 30

  * Max iterations: 1000

- Key Functions:

  * sigmoid(): Converts continuous values to binary (0/1)

  * velocity update: v = w*v + c1*r1*(pbest-x) + c2*r2*(gbest-x)

  * position update: x = 1 if rand < sigmoid(v) else 0

2. BINARY GSA (BGSA)

- Core Concept: Based on gravitational forces between masses

- Key Parameters:

  * G0: Initial gravitational constant (100)

  * alpha: Decay rate for G (20)

  * force_limit: Maximum allowed force (100)

  * Population size: 30

  * Max iterations: 1000

- Key Functions:
  * calculate_forces(): Computes gravitational forces between solutions
  * mass calculation: Based on fitness normalization
  * force update: F = G * (m1*m2/R) * direction

3. HYBRID PSO-GSA

- Core Concept: Combines exploration of GSA with exploitation of PSO
- Key Parameters:
  * PSO parameters (w, c1, c2)
  * GSA parameters (G0, alpha)
  * gsa_weight_start: 0.2 (initial GSA influence)
  * gsa_weight_end: 0.4 (final GSA influence)

- Key Functions:
  * Combined velocity update with weighting:
    ```python
    gsa_weight = gsa_weight_start + (gsa_weight_end - gsa_weight_start) * progress
    cognitive = c1 * r1 * (personal_best - positions)
    social = c2 * r2 * (global_best - positions)
    forces = calculate_forces(positions, fitness, G, t)
    ```

velocities = w*velocities + (1-gsa_weight)*(cognitive + social) + gsa_weight*forces

```

* This weighting scheme:

- Initially favors PSO (exploration)

- Gradually increases GSA influence (exploitation)

- Provides dynamic balance between algorithms

- Adapts search behavior over iterations

4. BINARY WHALE OPTIMIZATION (BWOA)

- Core Concept: Inspired by humpback whale hunting behavior

- Key Parameters:

* a: Decreases linearly from 2 to 0 (controls search range)

* b: Spiral shape constant (usually 1)

* p: Probability for hunting choice (0.5)

- Key Functions:

* Encircling prey: X = X* - A|CX* - X| where:

- X* is best solution position (prey location)

- X is current whale position

- A = 2a*r1 - a (r1 is random [0,1])

- C = 2*r2 (r2 is random [0,1])

- |A| < 1: move towards prey (exploitation)
```

- |A| > 1: search for better prey (exploration)

* Spiral update: X = D'ebl cos(2πl) + X*

* Search for prey: X = Xrand - A|CXrand - X|

Common Functions Across All Algorithms:

1. repair_solution():

- Purpose: Ensures solutions satisfy knapsack constraints

- Method:

* Checks if solution is feasible

* If infeasible, randomly removes items until constraints are met

* Returns feasible solution

2. evaluate_fitness():

- Purpose: Calculates solution quality

- Method:

* Computes total profit

* Applies penalty for constraint violations

* Returns final fitness value

3. sigmoid():

- Purpose: Converts continuous values to binary

- Method: 1/(1 + e^(-x))

- Used in position updates for all binary variants

4. optimize():

- Purpose: Main optimization loop

- Common steps:

  * Initialize population

  * Update positions/velocities

  * Apply repair mechanism

  * Track best solutions

  * Return best found solution

Key MKP Adaptations:

1. Binary representation for item selection

2. Constraint handling through repair mechanism

3. Fitness calculation incorporating penalties

4. Modified position updates for binary space

5. Parameter tuning for knapsack problem structure

The key innovations in each algorithm are:

- BPSO: Balance between individual and group knowledge

- BGSA: Mass-based search with gravitational forces

- Hybrid: Dynamic weighting between PSO and GSA behaviors using gsa_weight

- BWOA: Whale hunting behavior with adaptive search radius

These algorithms complement each other:

- PSO excels at fine-tuning solutions

- GSA provides good exploration

- Hybrid combines their strengths with adaptive weights

- BWOA offers unique search patterns based on prey hunting