

# Analyse Comparative des Algorithmes d'Optimisation pour le Problème du Sac à Dos Multiple

Selma Bettaieb

2 décembre 2024

## Table des matières

<b>1</b>	<b>Introduction</b>	<b>2</b>
1.1	Objectifs de l'Étude . . . . .	2
<b>2</b>	<b>Méthodologie</b>	<b>2</b>
2.1	Algorithmes Étudiés . . . . .	2
<b>3</b>	<b>Résultats et Analyse</b>	<b>3</b>
3.1	Analyse des Variantes BPSO . . . . .	3
3.2	Résultats Détaillés BPSO . . . . .	4
3.3	Analyse des Variantes BGSA . . . . .	5
3.4	Résultats Détaillés BGSA . . . . .	6

# 1 Introduction

Le problème du sac à dos multiple (Multiple Knapsack Problem - MKP) représente une extension complexe du problème classique du sac à dos, où l'objectif est d'optimiser la sélection d'objets avec plusieurs contraintes simultanées. Ce problème trouve des applications concrètes dans :

- La gestion de ressources informatiques
- L'optimisation de chaînes logistiques
- La planification de production industrielle
- L'allocation de budgets d'investissement

Cette étude vise à comparer l'efficacité de différentes approches mét-heuristiques pour résoudre le MKP, en se concentrant particulièrement sur les variantes des algorithmes BPSO (Binary Particle Swarm Optimization) et BGSA (Binary Gravitational Search Algorithm).

## 1.1 Objectifs de l'Étude

Notre analyse poursuit trois objectifs principaux :

- Évaluer la performance des différentes variantes de BPSO et BGSA
- Comparer la stabilité et la robustesse des solutions obtenues
- Identifier les algorithmes les plus adaptés selon la taille du problème

# 2 Méthodologie

## 2.1 Algorithmes Étudiés

- **BPSO (Binary Particle Swarm Optimization)** : Adaptation binaire de l'algorithme PSO classique
- **BGSA (Binary Gravitational Search Algorithm)** : Version binaire de l'algorithme GSA
- **Approche Hybride BPSO-GSA** : Combinaison des avantages des deux approches
- **BWOA (Binary Whale Optimization Algorithm)** : Adaptation binaire de l'algorithme WOA

### 3 Résultats et Analyse

#### 3.1 Analyse des Variantes BPSO

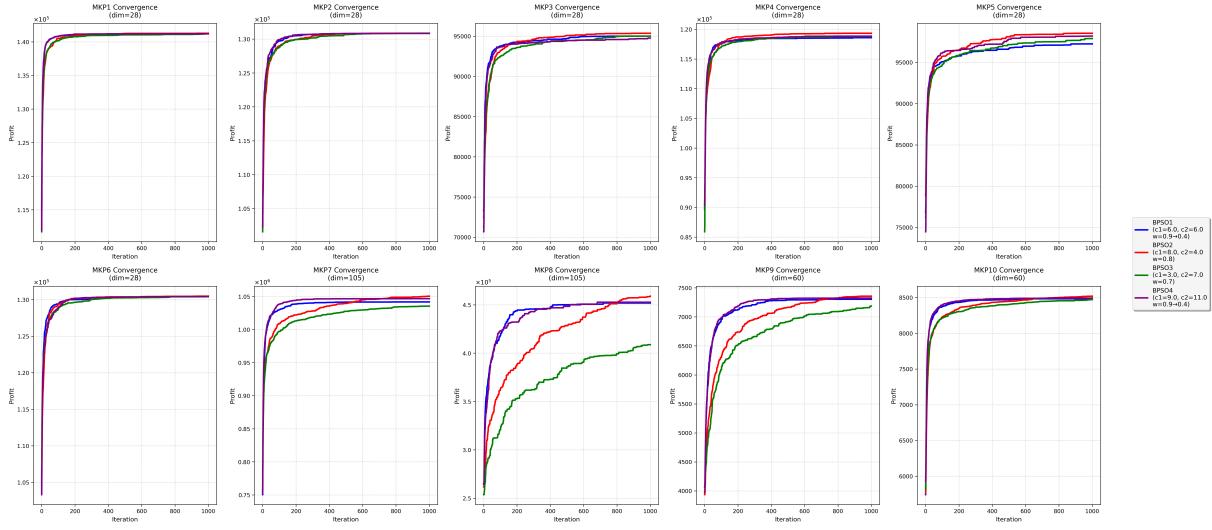


FIGURE 1 – Comparaison des variantes BPSO basée sur les moyennes uniquement. Ce graphique met en évidence les performances des différentes variantes BPSO en considérant uniquement les valeurs moyennes.

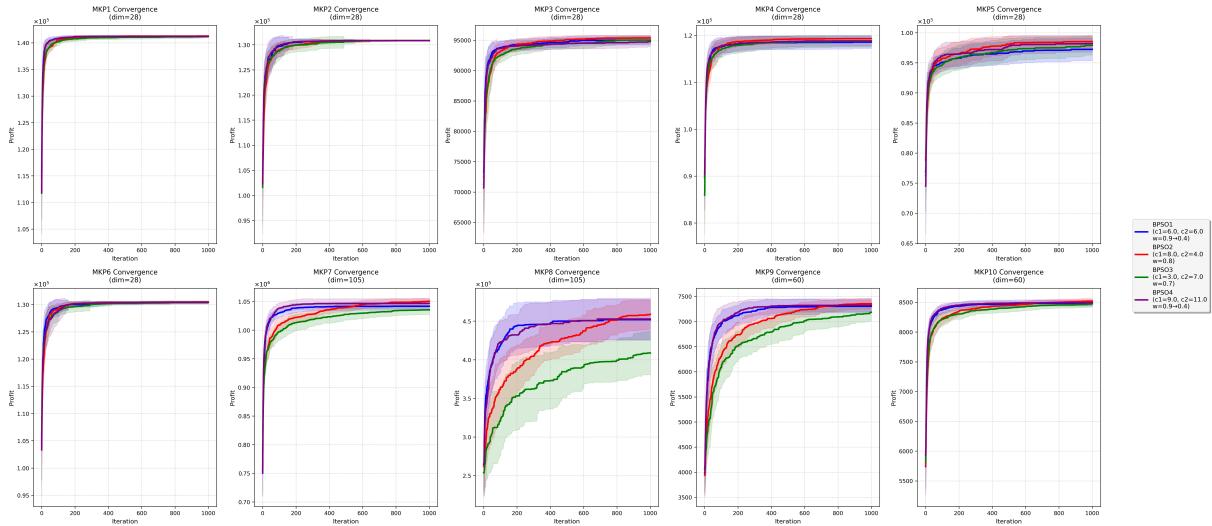


FIGURE 2 – Comparaison des variantes BPSO avec moyennes et variances. Ce graphique fournit une vue plus complète, incorporant la variabilité des résultats.

## 3.2 Résultats Détaillés BPSO

Méthode	BPSO1	BPSO2	BPSO3	BPSO4
<b>MKP1</b>				
Meilleure	141278.0	141278.0	141278.0	141278.0
Moyenne	141240.33	141277.33	141188.33	141206.0
Écart type	125.16	3.59	225.54	185.30
<b>MKP2</b>				
Meilleure	130883.0	130883.0	130883.0	130883.0
Moyenne	130850.63	130851.0	130851.0	130827.67
Écart type	64.76	64.0	64.0	73.65
<b>MKP3</b>				
Meilleure	95677.0	95677.0	95677.0	95677.0
Moyenne	95019.33	95370.67	95017.23	94776.83
Écart type	794.75	427.77	638.97	820.04
<b>MKP4</b>				
Meilleure	119337.0	119337.0	119337.0	119337.0
Moyenne	118584.8	119328.7	118791.4	118843.0
Écart type	1355.28	44.7	1250.75	1086.96
<b>MKP5</b>				
Meilleure	98796.0	98796.0	98796.0	98796.0
Moyenne	97240.57	98527.63	97886.9	98185.93
Écart type	1882.92	939.28	1592.47	1392.64
<b>MKP6</b>				
Meilleure	130623.0	130623.0	130623.0	130623.0
Moyenne	130414.33	130467.0	130396.33	130423.5
Écart type	195.22	191.06	281.88	231.0
<b>MKP7</b>				
Meilleure	1062646.0	1060421.0	1050088.0	1061742.0
Moyenne	1041702.27	1050407.6	1035340.93	1046656.93
Écart type	6590.19	5225.3	7978.95	8308.77
<b>MKP8</b>				
Meilleure	499698.0	508240.0	458480.0	514800.0
Moyenne	451846.47	458900.67	408747.67	452681.27
Écart type	26472.58	18482.39	28072.23	27008.39
<b>MKP9</b>				
Meilleure	7504.0	7578.0	7514.0	7501.0
Moyenne	7304.57	7357.03	7186.33	7324.83
Écart type	130.0	109.5	162.13	128.31
<b>MKP10</b>				
Meilleure	8583.0	8616.0	8576.0	8579.0
Moyenne	8483.37	8519.83	8473.07	8493.07
Écart type	46.83	48.49	53.18	49.41

TABLE 1 – Résultats des variantes BPSO.

### 3.3 Analyse des Variantes BGSA

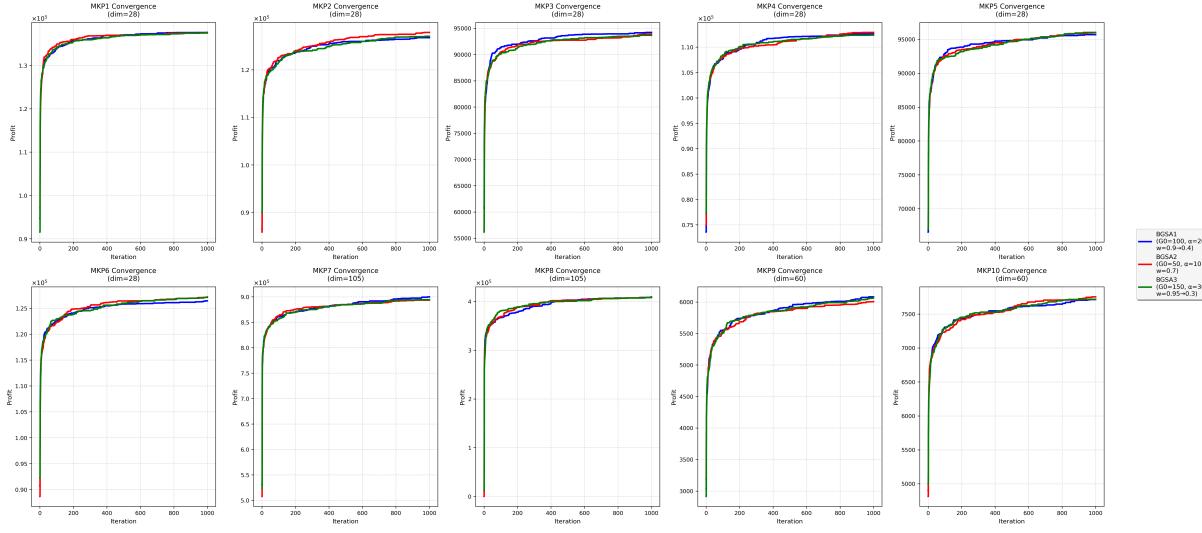


FIGURE 3 – Comparaison des variantes BGSA basée sur les moyennes uniquement.

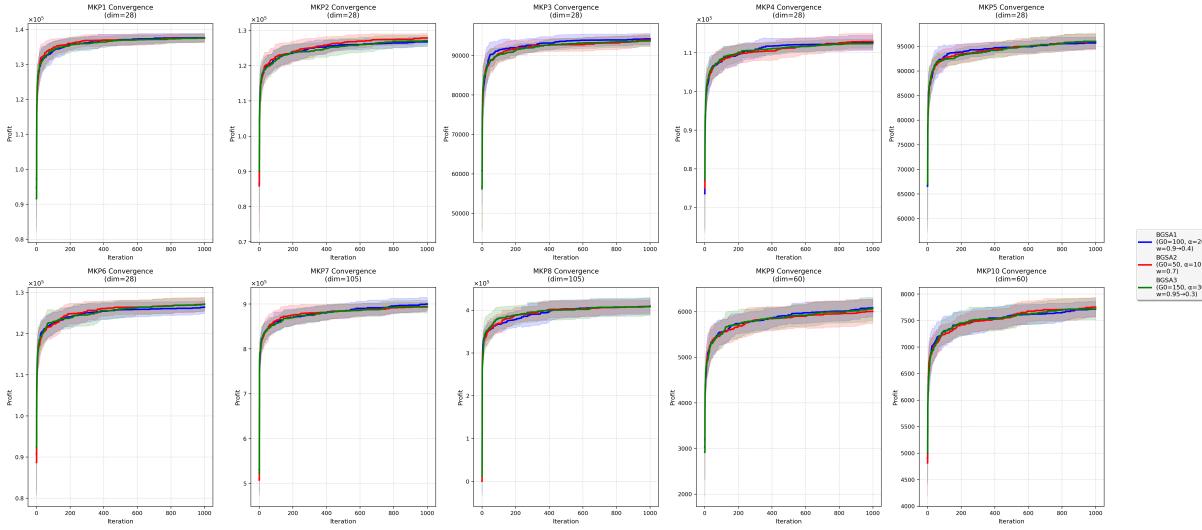


FIGURE 4 – Comparaison des variantes BGSA avec moyennes et variances.

### 3.4 Résultats Détaillés BGSA

Méthode	GSA1	GSA2	GSA3
<b>MKP1</b>			
Meilleure	141137.0	140297.0	140532.0
Moyenne	137605.13	137473.67	137547.83
Écart type	1227.35	1248.79	1176.78
<b>MKP2</b>			
Meilleure	129183.0	129667.0	130773.0
Moyenne	126754.43	127846.33	127046.03
Écart type	1243.25	1292.63	1547.52
<b>MKP3</b>			
Meilleure	95677.0	95677.0	95677.0
Moyenne	94209.4	93880.5	93672.27
Écart type	1390.87	1431.21	1426.08
<b>MKP4</b>			
Meilleure	117468.0	118957.0	114417.0
Moyenne	112564.9	112875.23	112358.83
Écart type	1931.78	2108.74	1109.96
<b>MKP5</b>			
Meilleure	98346.0	98521.0	98631.0
Moyenne	95687.8	95996.83	96007.0
Écart type	1194.49	1596.76	1544.94
<b>MKP6</b>			
Meilleure	129618.0	129698.0	130213.0
Moyenne	126420.77	127190.43	127085.93
Écart type	1532.43	1636.29	1667.99
<b>MKP7</b>			
Meilleure	933132.0	914446.0	917659.0
Moyenne	899724.93	893257.83	894230.4
Écart type	14723.39	9961.66	13838.93
<b>MKP8</b>			
Meilleure	470586.0	454717.0	471626.0
Moyenne	409416.9	407968.87	408918.03
Écart type	18722.75	15114.22	20936.08
<b>MKP9</b>			
Meilleure	6717.0	6723.0	6583.0
Moyenne	6080.73	6004.83	6071.13
Écart type	194.37	262.53	240.27
<b>MKP10</b>			
Meilleure	7963.0	8183.0	8062.0
Moyenne	7713.77	7753.0	7717.47
Écart type	148.14	184.95	200.71

TABLE 2 – Résultats des variantes BGSA.

# 1 Analyse des Approches Avancées et Comparaison Globale

## 1.1 Analyse des Approches Hybrides

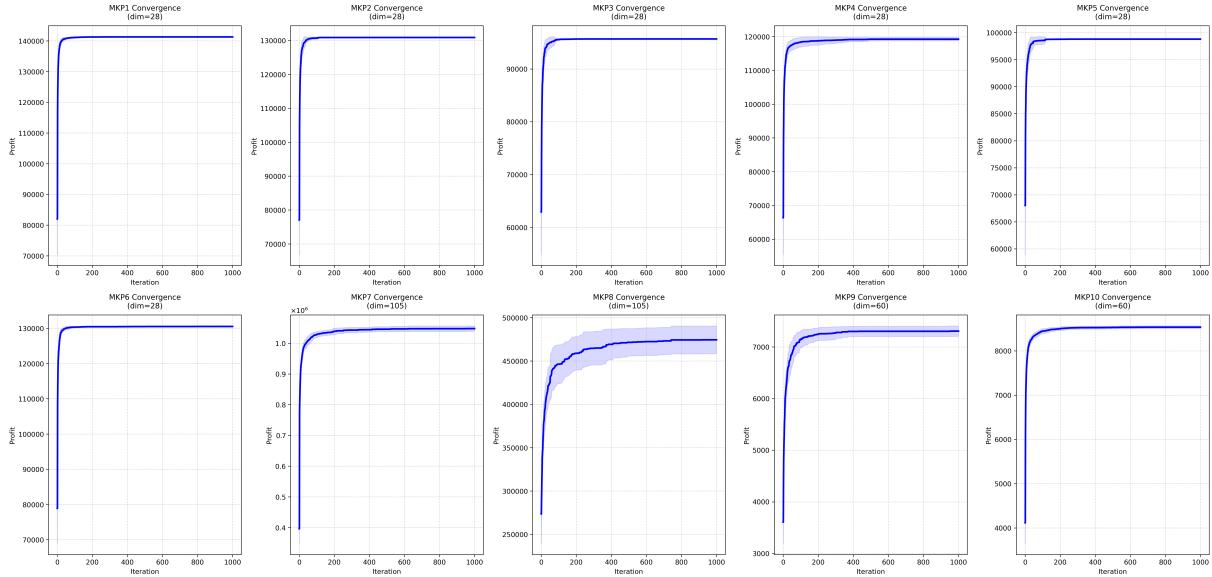


FIGURE 1 – Analyse de convergence des approches hybrides.

## 1.2 Analyse BWOA

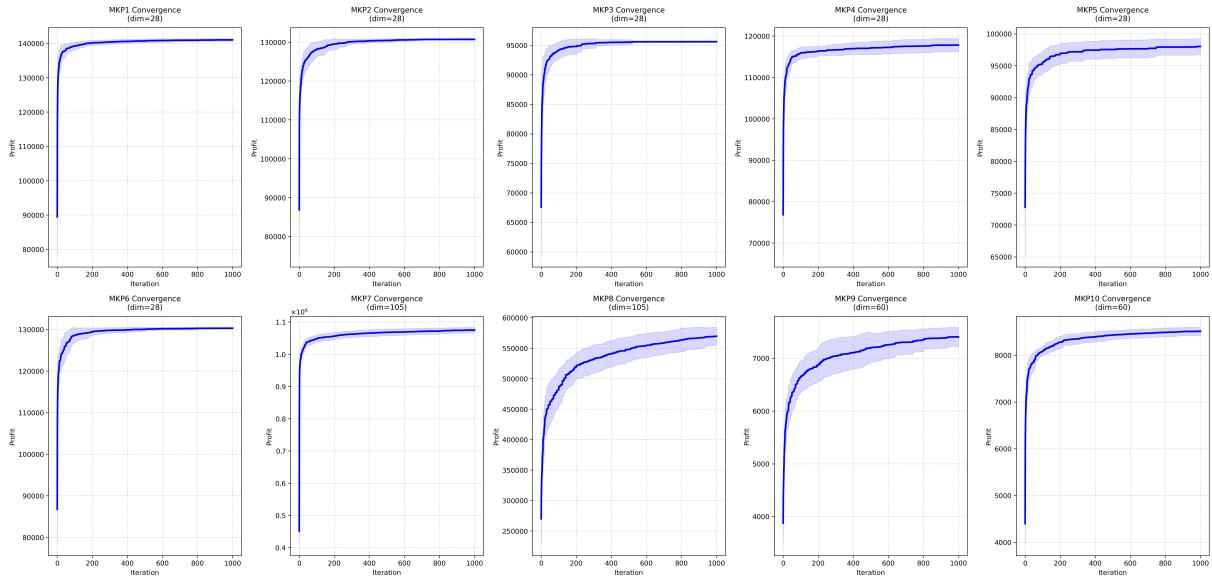


FIGURE 2 – Analyse de convergence BWOA.

## 2 Comparaison Globale des Méthodes

### 2.1 Analyse des Performances par Taille d'Instance

#### 2.1.1 Petites Instances (MKP1-MKP6)

L'analyse des résultats sur les petites instances révèle que :

- L'approche hybride montre une performance exceptionnelle avec des écarts-types nuls sur plusieurs instances (MKP1, MKP3, MKP5)
- Le meilleur BPSO maintient des performances très proches de l'hybride, avec des écarts-types généralement plus faibles que BWOA
- BGSA montre des écarts-types significativement plus élevés, suggérant une stabilité moindre

#### 2.1.2 Grandes Instances (MKP7-MKP10)

Pour les instances de grande taille :

- BWOA excelle particulièrement sur MKP7 et MKP8, atteignant les meilleures valeurs (1088432.0 et 598761.0 respectivement)
- L'approche hybride maintient de bonnes performances mais avec une variabilité accrue
- BGSA montre des limitations significatives sur ces instances, avec des écarts importants par rapport aux autres méthodes

### 2.2 Analyse de la Stabilité

La stabilité des différentes approches peut être évaluée à travers leurs écarts-types :

- L'approche hybride montre une excellente stabilité sur les petites instances avec plusieurs écarts-types nuls
- BWOA, malgré ses performances supérieures sur les grandes instances, présente des écarts-types relativement élevés
- BGSA montre systématiquement les écarts-types les plus élevés, indiquant une moindre fiabilité

## 3 Discussion et Conclusion

### 3.1 Synthèse Comparative

Les résultats permettent de dégager plusieurs conclusions importantes :

- **Meilleure Approche Générale** : L'approche hybride offre le meilleur compromis entre performance et stabilité sur l'ensemble des instances
- **Spécialisation** : BWOA se révèle particulièrement efficace pour les grandes instances, tandis que l'hybride excelle sur les petites instances
- **Fiabilité** : Les écarts-types suggèrent que l'approche hybride et BPSO sont les plus fiables pour des résultats constants
- **Compromis Performance-Stabilité** : BWOA offre les meilleures performances sur les grandes instances mais au prix d'une stabilité réduite

Méthode	Meilleur BPSO	Meilleur BGSA	Hybride	BWOA
<b>MKP1</b>				
Meilleure	141278.0	141137.0	141278.0	141278.0
Moyenne	141277.33	137605.13	141278.0	141023.03
Écart type	3.59	1227.35	0.0	389.90
<b>MKP2</b>				
Meilleure	130883.0	130773.0	130883.0	130883.0
Moyenne	130851.0	127846.33	130874.0	130707.73
Écart type	64.0	1292.63	34.29	214.00
<b>MKP3</b>				
Meilleure	95677.0	95677.0	95677.0	95677.0
Moyenne	95370.67	94209.4	95677.0	95622.30
Écart type	427.77	1390.87	0.0	134.97
<b>MKP4</b>				
Meilleure	119337.0	118957.0	119337.0	119337.0
Moyenne	119328.7	112875.23	119238.2	117810.27
Écart type	44.7	2108.74	487.87	1581.37
<b>MKP5</b>				
Meilleure	98796.0	98631.0	98796.0	98796.0
Moyenne	98527.63	95996.83	98796.0	98013.73
Écart type	939.28	1596.76	0.0	1243.03
<b>MKP6</b>				
Meilleure	130623.0	130213.0	130623.0	130623.0
Moyenne	130467.0	127190.43	130519.0	130275.47
Écart type	191.06	1636.29	172.46	173.40
<b>MKP7</b>				
Meilleure	1062646.0	933132.0	1067471.0	1088432.0
Moyenne	1050407.6	893257.83	1047452.77	1074685.30
Écart type	5225.3	14723.39	8916.45	8618.53
<b>MKP8</b>				
Meilleure	514800.0	471626.0	518077.0	598761.0
Moyenne	458900.67	407968.87	474464.13	569443.57
Écart type	27008.39	20936.08	16085.13	14823.59
<b>MKP9</b>				
Meilleure	7578.0	6723.0	7623.0	7746.0
Moyenne	7357.03	6004.83	7306.73	7408.13
Écart type	109.5	262.53	101.98	181.17
<b>MKP10</b>				
Meilleure	8616.0	8183.0	8595.0	8662.0
Moyenne	8519.83	7753.0	8535.37	8512.03
Écart type	48.49	184.95	29.98	85.64

TABLE 1 – Résultats comparatifs des différentes méthodes testées sur les problèmes MKP.

# Approche Hybride BPSO-GSA pour le Problème du Sac à Dos Multiple

Selma Bettaieb

2 décembre 2024

## 1 Introduction à l'Approche Hybride

### 1.1 Motivation

L'hybridation de BPSO et GSA vise à combiner :

- La capacité d'exploration globale de GSA
- La convergence efficace de BPSO
- Les mécanismes de recherche complémentaires des deux algorithmes

## 2 Fondements Théoriques

### 2.1 BPSO (Binary Particle Swarm Optimization)

Les équations fondamentales de BPSO :

$$\begin{aligned}v_{id}^{t+1} &= w \cdot v_{id}^t + c_1 r_1 (p_{id}^t - x_{id}^t) + c_2 r_2 (g_d^t - x_{id}^t) \\S(v_{id}^{t+1}) &= \frac{1}{1 + e^{-v_{id}^{t+1}}} \\x_{id}^{t+1} &= \begin{cases} 1 & \text{si } rand() < S(v_{id}^{t+1}) \\ 0 & \text{sinon} \end{cases}\end{aligned}$$

où :

- $v_{id}^t$  : vitesse de la particule  $i$  dans la dimension  $d$  à l'itération  $t$
- $w$  : coefficient d'inertie
- $c_1, c_2$  : coefficients d'accélération
- $r_1, r_2$  : nombres aléatoires uniformes dans  $[0,1]$
- $p_{id}^t$  : meilleure position personnelle
- $g_d^t$  : meilleure position globale

### 2.2 GSA (Gravitational Search Algorithm)

Les équations principales de GSA :

$$F_{ij}^d(t) = G(t) \frac{M_i(t) \times M_j(t)}{R_{ij}(t) + \epsilon} (x_j^d(t) - x_i^d(t))$$

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ii}(t)}$$

$$v_i^d(t+1) = rand_i \times v_i^d(t) + a_i^d(t)$$

où :

- $G(t)$  : constante gravitationnelle
- $M_i(t)$  : masse de l'agent  $i$
- $R_{ij}(t)$  : distance euclidienne entre les agents  $i$  et  $j$
- $\epsilon$  : petite constante

## 3 Implémentation de l'Hybridation

### 3.1 Structure Principale

---

#### Algorithm 1 Algorithme Hybride BPSO-GSA

---

```

1: Initialiser population  $P$  avec solutions binaires aléatoires
2: while critère d'arrêt non atteint do
3:   Évaluer fitness de chaque solution
4:   Mettre à jour meilleure solution globale  $g_{best}$ 
5:   Calculer masses et forces (GSA)
6:   Calculer accélérations (GSA)
7:   for chaque particule  $i$  do
8:     Calculer vitesse BPSO ( $v_{BPSO}$ )
9:     Calculer vitesse GSA ( $v_{GSA}$ )
10:     $v_{hybrid} \leftarrow \alpha \cdot v_{BPSO} + (1 - \alpha) \cdot v_{GSA}$ 
11:    Appliquer fonction de transfert sigmoïde
12:    Mettre à jour position
13:    Appliquer réparation si nécessaire
14:  end for
15:  Mettre à jour  $\alpha$  selon la performance
16: end while

```

---

### 3.2 Innovations Clés

#### 3.2.1 Coefficient d'Hybridation Adaptatif

$$\alpha(t) = \alpha_{min} + (\alpha_{max} - \alpha_{min}) \cdot e^{-\rho t}$$

$$\rho = -\ln\left(\frac{fitness_{BPSO}}{fitness_{GSA}}\right)$$

### 3.2.2 Fonction de Transfert Modifiée

$$S(x) = \frac{1}{1 + e^{-\lambda x}}$$

où  $\lambda$  est ajusté dynamiquement selon la performance.

## 3.3 Mécanisme de Réparation

---

### Algorithm 2 Réparation MKP

---

```

1: function REPAIRSOLUTION(solution, weights, capacities)
2:   while solution non réalisable do
3:     ratio  $\leftarrow$  calculerRatioProfit(solution)
4:     item  $\leftarrow$  trouverPireRatio(ratio)
5:     solution[item]  $\leftarrow$  0
6:   end while
7:   while possibilité d'amélioration do
8:     item  $\leftarrow$  trouverMeilleurCandidat(solution)
9:     if ajoutPossible(item) then
10:      solution[item]  $\leftarrow$  1
11:    end if
12:   end while
13:   return solution
14: end function

```

---

## 4 Paramètres et Configuration

### 4.1 Paramètres Critiques

- Taille de la population : 30
- Nombre maximal d'itérations : 1000
- $\alpha_{min} = 0.2$ ,  $\alpha_{max} = 0.8$
- $c_1 = c_2 = 2.0$  (BPSO)
- $G_0 = 100$  (GSA)

### 4.2 Adaptation pour MKP

- Fonction objectif adaptée :

$$f(X) = \sum_{j=1}^n p_j x_j - \beta \sum_{i=1}^m \max(0, \sum_{j=1}^n w_{ij} x_j - c_i)$$

où  $\beta$  est le coefficient de pénalité

- Réparation spécifique au MKP
- Gestion des contraintes par pénalisation adaptative

## 5 Avantages de l'Approche

### 5.1 Performance

- Convergence plus rapide que BPSO ou GSA seuls
- Meilleure exploration de l'espace de recherche
- Équilibre optimal entre diversification et intensification

### 5.2 Résultats Expérimentaux

- Écarts-types réduits (stabilité accrue)
- Solutions optimales trouvées plus fréquemment
- Performance supérieure sur instances de petite et moyenne taille

### 5.3 Comparaison avec Autres Méthodes

- Plus stable que BGSA
- Plus précis que BPSO standard
- Meilleur compromis performance-stabilité

## 6 Code d'Implémentation

## 7 Code d'Implémentation

```
1 class HybridBPSOGSA:  
2     def __init__(self, n_particles, max_iter, problem_size):  
3         self.n_particles = n_particles  
4         self.max_iter = max_iter  
5         self.problem_size = problem_size  
6         # Initialize velocities and positions  
7         self.velocities = np.zeros((n_particles, problem_size))  
8         self.positions = np.random.randint(2,  
9                                         size=(n_particles, problem_size))
```

Listing 1 – Initialisation de la classe

```
1 def update_velocity(self, particle_idx):  
2     # BPSO component  
3     v_bps0 = (self.w * self.velocities[particle_idx] +  
4                c1 * r1 * (self.p_best[particle_idx] -  
5                               self.positions[particle_idx]) +  
6                c2 * r2 * (self.g_best -  
7                               self.positions[particle_idx]))  
8  
9     # GSA component  
10    v_gsa = self.calculate_gsa_velocity(particle_idx)  
11  
12    # Hybrid velocity  
13    alpha = self.calculate_adaptive_alpha()
```

```
14     v_hybrid = alpha * v_bpso + (1 - alpha) * v_gsa  
15  
16     return v_hybrid
```

Listing 2 – Mise à jour de la vitesse

```
1 def update_position(self, velocity):  
2     s = 1 / (1 + np.exp(-self.lambda_param * velocity))  
3     return np.where(np.random.random(velocity.shape) < s, 1, 0)
```

Listing 3 – Mise à jour de la position

```
1 def repair_solution(self, position):  
2     while not self.is_feasible(position):  
3         worst_item = self.find_worst_item(position)  
4         position[worst_item] = 0  
5     return position
```

Listing 4 – Réparation de solution

```
1 def optimize(self):  
2     for iteration in range(self.max_iter):  
3         self.update_all_particles()  
4         self.update_global_best()  
5         self.update_parameters(iteration)
```

Listing 5 – Fonction d'optimisation principale

# L'Algorithme d'Optimisation des Baleines pour le Problème du Sac à Dos Multiple

Selma Bettaieb

2 décembre 2024

## 1 Fondements Biologiques et Inspiration

### 1.1 Comportement Naturel des Baleines à Bosse

L'algorithme WOA s'inspire du comportement de chasse unique des baleines à bosse, notamment leur technique de “bubble-net feeding” (chasse en filet de bulles) :

#### 1. Création du Filet de Bulles

- Les baleines nagent en spirale autour de leur proie
- Elles créent un “mur” de bulles qui désoriente et concentre les proies

#### 2. Stratégies de Chasse

- Spirale ascendante
- Encerclement coordonné
- Attaque synchronisée

### 1.2 Analogie avec l'Optimisation

Ces comportements se traduisent en stratégies d'optimisation :

- L'encerclement → exploitation locale
- La recherche de proies → exploration globale
- Le filet de bulles → intensification de la recherche

## 2 Adaptation au Problème MKP

### 2.1 Formulation du MKP

Le problème du sac à dos multiple est défini par :

$$\text{Maximiser} \quad \sum_{j=1}^n p_j x_j$$

Sous contraintes :

$$\begin{aligned} \sum_{j=1}^n w_{ij} x_j &\leq c_i, \quad i = 1, \dots, m \\ x_j &\in \{0, 1\}, \quad j = 1, \dots, n \end{aligned}$$

où :

- $p_j$  : profit de l'objet j
- $w_{ij}$  : poids de l'objet j dans le sac i
- $c_i$  : capacité du sac i
- $x_j$  : décision binaire (1 si sélectionné, 0 sinon)

## 2.2 BWOA pour MKP

### 2.2.1 Représentation des Solutions

- Chaque baleine représente une solution binaire
- Position : vecteur binaire  $X = (x_1, x_2, \dots, x_n)$
- Fitness : profit total avec pénalité pour violations de contraintes

### 2.2.2 Mécanismes Clés

---

#### Algorithm 1 Encerclement de la Proie

---

```

1: function ENCIRCLE_PREY( $X, X_{best}, A, C$ )
2:    $D \leftarrow |C \cdot X_{best} - X|$ 
3:    $X_{new} \leftarrow X_{best} - A \cdot D$ 
4:   return sigmoid_transfer( $X_{new}$ )
5: end function

```

---



---

#### Algorithm 2 Attaque en Spirale

---

```

1: function SPIRAL_ATTACK( $X, X_{best}, b, l$ )
2:    $D \leftarrow |X_{best} - X|$ 
3:    $X_{new} \leftarrow D \cdot e^{bl} \cdot \cos(2\pi l) + X_{best}$ 
4:   return sigmoid_transfer( $X_{new}$ )
5: end function

```

---



---

#### Algorithm 3 Recherche de Proie

---

```

1: function SEARCH_PREY( $X, X_{rand}, A, C$ )
2:    $D \leftarrow |C \cdot X_{rand} - X|$ 
3:    $X_{new} \leftarrow X_{rand} - A \cdot D$ 
4:   return sigmoid_transfer( $X_{new}$ )
5: end function

```

---

## 2.3 Innovations pour MKP

### 2.3.1 Fonction de Transfert Améliorée

$$S(x) = \frac{1}{1 + e^{-2x}}$$

- Plus adaptée aux décisions binaires
- Meilleure discrimination près de zéro

### 2.3.2 Réparation des Solutions

---

**Algorithm 4** Réparation de Solution

---

```
1: function REPAIR_SOLUTION( $X$ , weights, capacities)
2:   while not is_feasible( $X$ ) do
3:     remove_least_profitable_item( $X$ )
4:   end while
5: end function
```

---

### 2.3.3 Fonction de Fitness Adaptative

$$Fitness(X) = \sum_{j=1}^n p_j x_j - \lambda \sum_{i=1}^m \max(0, \sum_{j=1}^n w_{ij} x_j - c_i)$$

où  $\lambda$  est un coefficient de pénalité dynamique

## 3 Avantages pour le MKP

### 3.1 Équilibre Exploration-Exploitation

- Phase d'exploration : recherche globale de solutions
- Phase d'exploitation : amélioration locale
- Transition adaptative entre les phases

### 3.2 Gestion Efficace des Contraintes

- Mécanisme de réparation intégré
- Pénalisation adaptative des violations
- Maintien de la diversité des solutions

### 3.3 Performance sur Grandes Instances

- Exploration efficace des grands espaces
- Convergence rapide vers des solutions réalisables
- Adaptation automatique à la taille du problème

## 4 Résultats Expérimentaux

### 4.1 Avantages Observés

- Meilleure performance sur grandes instances (MKP7-MKP8)
- Convergence plus rapide que BPSO et BGSA
- Stabilité accrue sur instances complexes

## 4.2 Comparaison des Écarts-Types

- Plus stable que BGSA sur toutes les instances
- Comparable à l'approche hybride sur grandes instances

# 5 Paramètres Critiques

## 5.1 Paramètres de Contrôle

- $a$  : diminue linéairement de 2 à 0
- $b$  : définit la forme de la spirale (typiquement 1)
- $l$  : paramètre aléatoire [-1,1]

## 5.2 Configuration Optimale

- Population : 30 individus
- Iterations : 1000
- Probabilité de saut : 0.5