**RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE**
**MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE**
**UNIVERSITÉ ABDERRAHMANE MIRA DE BÉJAIA**

جامعة بجاية
Tasdawit n Bgayet
Université de Béjaïa

**FACULTÉ DES SCIENCES EXACTES**
**DÉPARTEMENT D'INFORMATIQUE**

**MÉMOIRE DE MASTER RECHERCHE**
**INFORMATIQUE**
**OPTION : SYSTÈMES D'INFORMATION AVANCÉS**

## Thème :

# Placement intelligent des Conteneurs dans un Terminal Portuaire

**Présenté par :**
**SMAILI Billal et DELLIDJ Anis**

**Encadreur académique :** Samira AIT KACI AZZOU Epse BOUKERRAM
**Co-encadrante :** Djamila BOUKREDERA
**Directeur de stage :** Belal BEDAOUCHE

Soutenu le **01 juillet 2024 à 8h30** devant le jury composé de :

| | |
|---|---|
| **Présidente** | : Karima ADEL |
| **Examinateur** | : Slimane GOUDJIL |
| **Examinateur** | : Khaled ALLEM |

**(Promotion 2023 - 2024)**

# Table des matières

# Table des figures

# Liste des tableaux

# *Remerciements*

*Avant tout, nous tenons à exprimer notre profonde gratitude envers Dieu, le Tout-Puissant, pour nous avoir donné la force et la persévérance nécessaires à l'accomplissement de ce travail.*

*Nous exprimons notre sincère reconnaissance à notre encadreur académique,* ***Madame Samira AIT KACI AZZOU Epse BOUKERRAM****, pour sa disponibilité, ses précieux conseils et son soutien constant tout au long de ce projet. Ses encouragements et son expertise nous ont guidés et permis de mener à bien ce mémoire.*

*Nos sincères remerciements à* ***Madame Adel Karima****, présidente du jury, pour avoir présidé notre soutenance et pour l'intérêt qu'elle a porté à notre travail.*

*Nous tenons également à remercier sincèrement nos examinateurs,* ***Monsieur Allem Khaled*** *et* ***Monsieur Goudjil Slimane****, pour leur temps précieux consacré à l'évaluation de notre travail et pour leurs remarques constructives.*

*Nos remerciements s'adressent également à notre co-encadrante,* ***Madame Djamila Boukredera****, pour sa contribution et ses conseils avisés qui ont enrichi notre recherche.*

*Enfin, nous remercions chaleureusement nos familles et nos amis pour leur soutien indéfectible et leur patience tout au long de ce parcours.*

# Introduction Générale

Le transport maritime, pilier indiscutable du commerce international, assure la circulation de plus de 80% des biens échangés à travers le globe. Au cœur de ce réseau complexe, les terminaux à conteneurs jouent un rôle crucial, agissant comme des plateformes logistiques dynamiques où transitent des millions de conteneurs chaque année. La fluidité et l'efficacité des opérations au sein de ces terminaux sont des facteurs déterminants pour la compétitivité des ports, la fiabilité des chaînes d'approvisionnement et la satisfaction des clients. Parmi les nombreux processus qui régissent le fonctionnement d'un terminal à conteneurs, le **placement stratégique des conteneurs** dans le yard de stockage revêt une importance particulière. Une organisation spatiale adéquate des conteneurs permet non seulement d'optimiser l'utilisation de l'espace disponible, mais aussi de fluidifier les opérations de manutention et de minimiser les déplacements inutiles. Face à l'essor continu du trafic maritime et à la diversité croissante des types de conteneurs, les terminaux font face à des défis de plus en plus complexes : congestion, saturation de l'espace, et difficultés à respecter les délais de livraison. Les port, acteurs majeur du commerce extérieur algérien, sont confronté à ces enjeux. La croissance du trafic conteneurisé, combinée à des contraintes d'espace et d'infrastructures, a entraîné des situations de congestion, des temps d'attente prolongés pour les navires et des difficultés à garantir des livraisons rapides et fiables. Améliorer la gestion du placement des conteneurs dans les Ports est donc une priorité pour renforcer leur performance et leur compétitivité. Ce mémoire propose une approche innovante pour répondre à cette problématique. Au lieu de se baser uniquement sur des règles empiriques ou des méthodes mathématiques traditionnelles, ce travail explore une **solution hybride** combinant un **modèle de prédiction de la priorité des conteneurs** et des **algorithmes de placement intelligents**. L'objectif est d'anticiper la date de sortie des conteneurs et de les organiser spatialement de manière à faciliter leur accès, à fluidifier les opérations de manutention et à maximiser l'utilisation de l'espace disponible. Pour atteindre cet objectif, ce travail s'articule autour des étapes suivantes :

1. **Analyse approfondie des processus de stockage dans les ports** : Identification des flux de marchandises, des types de conteneurs, des contraintes d'espace et des pratiques actuelles de placement.
2. **Conception d'un modèle de prédiction de la priorité des conteneurs** : Utilisation d'algorithmes d'apprentissage automatique pour estimer la durée de stockage des conteneurs en fonction de caractéristiques pertinentes.
3. **Implémentation d'algorithmes de placement intelligents** : Adaptation des algorithmes de recherche exaustive, Biogeography-Based Optimization (BBO) et Algorithme Génétique (AG) pour tenir compte des prédictions des priorités et prioriser les conteneurs à sortie rapide.

Ce mémoire s'articule autour de trois chapitres :

FIGURE 1 – Congestion du trafic dans le yard de stockage du Port de Bejaia

1. **Chapitre 1 :** Présentation du contexte général du transport maritime et des terminaux à conteneurs, des enjeux liés au placement des conteneurs et des solutions traditionnelles.
2. **Chapitre 2 :** Exploration de l'état de l'art des solutions de détection de l'emplacement des conteneurs, et des techniques d'apprentissage automatique utilisées dans les opérations portuaire.
3. **Chapitre 3 :** Présentation de la méthodologie adoptée pour ce travail, description de l'implémentation des algorithmes de placement et du modèle de prédiction de la priorité du conteneur, et présentation des résultats des simulations.

Et ce termine par une conclusion générale

# Chapitre 1 : Problématique du Placement des Conteneurs et Solutions Existantes

## 1.1 Introduction

Dans ce chapitre nous explorons les défis rencontrés dans le stockage des conteneurs, les méthodes traditionnelles de placement, et l'évolution vers l'utilisation de modèles mathématiques et d'apprentissage automatique. Nous examinerons comment ces avancées permettent de relever les défis de la gestion des placements des conteneurs et améliorer l'efficacité globale des terminaux à conteneurs.

## 1.2 Le transport maritime et les terminaux à conteneurs : Un contexte global

Le transport maritime est la pierre angulaire du commerce international, assurant la circulation de plus de 80% des marchandises échangées à l'échelle mondiale. Cette prédominance s'explique par plusieurs facteurs :

— **Capacité de transport** : Les navires porte-conteneurs modernes peuvent transporter des dizaines de milliers de conteneurs, offrant une capacité de chargement inégalée par les autres modes de transport. Cette capacité massive permet de réduire les coûts de transport par unité, rendant le transport maritime particulièrement attractif pour les marchandises volumineuses ou lourdes.

— **Connectivité globale** : Le réseau maritime mondial est extrêmement étendu et interconnecté, reliant les continents et les pays à travers un maillage dense de routes maritimes. Cette connectivité permet d'acheminer des marchandises vers pratiquement n'importe quelle destination dans le monde.

— **Coûts compétitifs** : Le transport maritime est généralement le mode de transport le plus économique pour les marchandises transportées sur de longues distances. Les coûts de carburant et de main-d'œuvre sont généralement plus faibles que pour le transport aérien ou terrestre, ce qui rend le transport maritime particulièrement attractif pour les produits à faible valeur ajoutée.

Les terminaux à conteneurs, en tant qu'interfaces entre le transport maritime et les modes de transport terrestres, jouent un rôle crucial dans la chaîne logistique mondiale. Ils constituent des plateformes logistiques essentielles pour la réception, le stockage, le transbordement et la livraison des conteneurs. Voici quelques exemples concrets illustrant l'importance du transport maritime et des terminaux à conteneurs pour l'économie algérienne :

— **Importation de biens de consommation** : L'Algérie importe une large gamme de produits manufacturés (véhicules, appareils électroniques, produits alimentaires) en provenance de différents pays du monde. Ces marchandises arrivent majoritairement par voie maritime, transitant par des terminaux à conteneurs tels que le Port d'Alger, le Port d'Oran ou le Port de Bejaia.

— **Exportation d'hydrocarbures** : L'Algérie est un important producteur et exportateur de pétrole et de gaz naturel. Ces hydrocarbures sont transportés par voie maritime vers des marchés internationaux, en utilisant des terminaux pétroliers et gaziers. Le complexe portuaire de Bejaia, par exemple, est un point de transit important pour les exportations d'hydrocarbures.

— **Développement du commerce régional** : Les terminaux à conteneurs jouent un rôle essentiel dans le développement du commerce régional en Algérie. Le Port de Bejaia, par exemple, dessert non seulement la région de Kabylie, mais aussi les wilayas limitrophes, favorisant ainsi les échanges commerciaux et l'intégration économique.

En résumé, le transport maritime et les terminaux à conteneurs sont des éléments fondamentaux de l'économie mondiale et jouent un rôle vital dans la facilitation des échanges commerciaux, le développement économique et la connectivité des pays.

## 1.3 Processus clés d'un terminal à conteneurs : Déchargement, stockage, chargement et livraison

Un terminal à conteneurs est une plateforme logistique complexe qui orchestre une série de processus interdépendants pour assurer la fluidité du transit des marchandises. Ces processus clés peuvent être décomposés en quatre phases principales :

1. **Déchargement des conteneurs** : Le déchargement des conteneurs consiste à transférer des conteneurs d'un navire à des moyens de transport terrestres ou à des aires de stockage. Ce processus débute par l'amarrage du navire au quai, suivi de l'utilisation de grues pour soulever les conteneurs et les déposer sur des chariots ou des camions. Ces conteneurs sont ensuite déplacés vers des zones de stockage temporaire où ils sont triés selon leur destination.

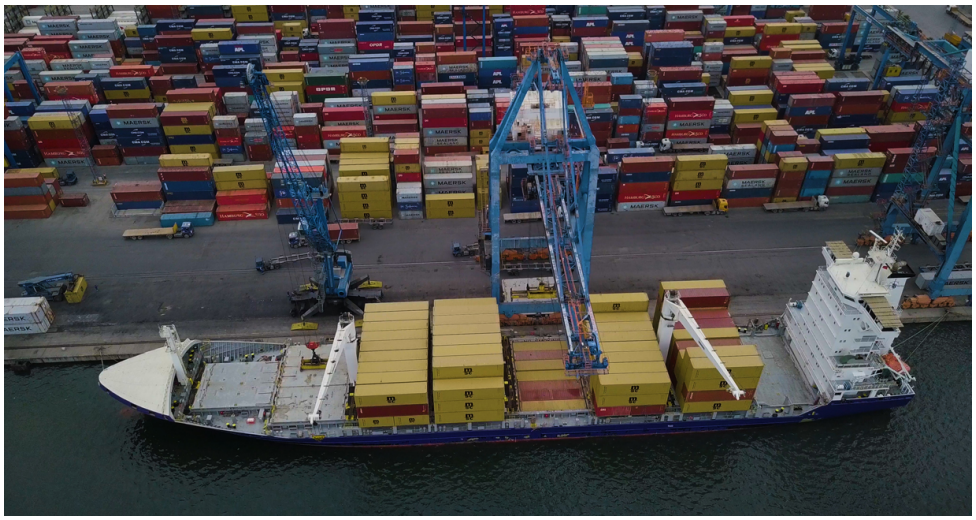


FIGURE 1.1 – Opération de déchargement des conteneurs

2. **Stockage des conteneurs** : Le stockage des conteneurs consiste à organiser et gérer les conteneurs dans des zones de dépôt temporaire ou permanente au sein d'un port ou d'une installation de stockage. Une fois déchargés des navires, les conteneurs sont transportés vers des aires de stockage spécifiques à l'aide de chariots élévateurs, de camions ou d'équipements spécialisés. Ils sont ensuite triés et classés selon des critères tels que la destination, le type de marchandise et les exigences de manipulation.
3. **Chargement des conteneurs** :Le chargement des conteneurs consiste à transférer des marchandises dans des conteneurs pour leur transport ultérieur par voie maritime, ferroviaire ou routière. Ce processus commence par la réception des marchandises au port ou à l'installation de chargement, où elles sont inspectées et préparées pour l'emballage. Les marchandises sont ensuite placées à l'intérieur des conteneurs de manière méthodique pour maximiser l'espace et minimiser les risques de dommages pendant le transport.
4. **Livraison des conteneurs** : La livraison des conteneurs consiste à transporter les conteneurs depuis leur point de stockage ou leur port d'arrivée jusqu'à leur destination finale. La figure 1.2 illustre la livraison d'un conteneur par voie terrestre.


FIGURE 1.2 – Livraison par voie terrestre

Ces processus, bien que distincts, sont étroitement interdépendants. Leur bon fonctionnement dépend d'une coordination efficace entre les différents acteurs du terminal (opérateurs de grues, conducteurs de véhicules, planificateurs, agents de douane, etc.) et d'une gestion optimale des ressources (espace de stockage, équipements de manutention).

## 1.4 Approches traditionnelles du placement des conteneurs

Pendant longtemps, le placement des conteneurs dans les terminaux portuaires a reposé sur des méthodes traditionnelles, fruit de l'expérience des opérateurs et des pratiques établies au fil du temps. Ces méthodes, souvent simples à mettre en œuvre, visaient à organiser le yard de stockage de manière intuitive et à faciliter les opérations de manutention. Cependant, l'essor du commerce international, la diversification des types de conteneurs et l'augmentation des volumes traités ont progressivement révélé les limites de ces approches traditionnelles.

Examinons plus en détail quelques exemples de méthodes traditionnelles couramment utilisées :

— **Placement par destination :**
Le placement par destination des conteneurs consiste à regrouper les conteneurs destinés à une même région géographique, à un pays spécifique ou à un client particulier dans des zones dédiées du yard de stockage. Cette méthode vise à optimiser les opérations logistiques en facilitant le chargement des conteneurs sur les navires.

— **Placement par type de conteneur :**
Le placement par type de conteneur dans les terminaux portuaires est une pratique visant à optimiser l'efficacité opérationnelle en regroupant les conteneurs selon leurs caractéristiques spécifiques. Cette méthode implique de séparer les

conteneurs en fonction de leur taille, de leur contenu ou de leur utilisation particulière dans des zones dédiées du yard de stockage.

— **Placement « Premier entré, dernier sorti » (FIFO) :**
Le principe du "Premier entré, dernier sorti" (FIFO) dans le placement des conteneurs dans les terminaux portuaires se réfère à une méthode où les conteneurs qui arrivent en premier sont stockés de manière à être accessibles en dernier pour le déchargement ou l'embarquement. Cette approche est souvent utilisée pour maximiser l'utilisation de l'espace disponible et minimiser les mouvements inutiles des conteneurs.

— **Placement « Emplacement vide le plus proche » :** Le placement des conteneurs selon le principe « Emplacement vide le plus proche » dans les terminaux portuaires repose sur la stratégie de placer les conteneurs dans les espaces disponibles les plus proches du point de déchargement initial, réduisant ainsi les distances parcourues par les équipements de manutention lors des premières étapes du processus logistique, de la quai vers le yard.

L'application de ces règles, si elle peut paraître simple et intuitive, se heurte à des limites importantes dans un contexte de terminaux à conteneurs modernes :

— **Manque d'anticipation** : Les règles empiriques ne prennent pas en compte les prévisions d'arrivée et de sortie des conteneurs.

— **Risque accru de relocations** : Le placement basé sur des critères simples comme la destination ou le type de conteneur ne garantit pas un accès facile aux conteneurs à sortie rapide.

— **Sous-utilisation de l'espace** : L'espace disponible dans le yard n'est pas exploité de manière optimale.

Face à ces limitations, les terminaux à conteneurs se tournent de plus en plus vers des approches plus sophistiquées, basées sur des modèles mathématiques et des algorithmes intelligents, pour améliorer la gestion du placement des conteneurs.

## 1.5 Méthodes mathématiques : Entre théorie et pratique

Face aux limites des règles traditionnelles, les terminaux à conteneurs se sont tournés vers des approches plus rigoureuses et systématiques, basées sur des modèles mathématiques. Ces méthodes visent à formuler le problème du placement des conteneurs comme un problème d'optimisation, en définissant une fonction objectif et un ensemble de contraintes. L'objectif peut être de minimiser le nombre de relocations, de maximiser l'utilisation de l'espace, ou d'autres critères pertinents pour l'efficacité opérationnelle du terminal.

Ces modèles mathématiques permettent de prendre en compte des variables complexes telles que les dates de sortie des conteneurs, les types de conteneurs, et la configuration du yard. Ils offrent une approche plus précise et automatisée pour le placement des conteneurs, comparée aux méthodes traditionnelles empiriques. En utilisant des techniques d'optimisation avancées, les terminaux peuvent mieux gérer les défis posés par la diversité des flux de conteneurs et les exigences croissantes en termes d'efficacité et de rentabilité dans le secteur de la logistique portuaire.

### 1.5.1 Difficultés d'application

Malgré son potentiel théorique, la programmation linéaire se heurte à des difficultés pratiques lorsqu'elle est appliquée au placement des conteneurs dans des terminaux réels :

- **Complexité des contraintes** : Les terminaux à conteneurs modernes sont soumis à un grand nombre de contraintes opérationnelles (ex : types de conteneurs, restrictions d'empilage, contraintes d'accès, dates de sortie fluctuantes), difficiles à modéliser de manière exhaustive avec des équations linéaires.
- **Taille des problèmes** : La taille des problèmes de placement des conteneurs dans les grands terminaux peut être considérable, avec des milliers de conteneurs à placer dans des yards de stockage vastes et complexes. La résolution de modèles de programmation linéaire de cette taille peut nécessiter des temps de calcul importants et des ressources informatiques considérables.
- **Dynamisme de l'environnement** : Les opérations dans un terminal à conteneurs sont soumises à des changements constants et imprévisibles (ex : arrivées tardives de navires, modifications des dates de sortie des conteneurs, variations du trafic). Ces changements rendent difficile la construction d'un modèle de programmation linéaire statique qui puisse refléter fidèlement la réalité opérationnelle du terminal.

## 1.6 Vers des solutions intelligentes : Intégration de l'apprentissage automatique

L'essor de l'apprentissage automatique et des technologies de l'information ouvre de nouvelles perspectives pour améliorer la gestion du placement des conteneurs dans les terminaux portuaires.L'intégration de ce dernier dans les opérations portuaires représente une avancée majeure pour optimiser la gestion des flux de conteneurs. En utilisant des modèles informatiques sophistiqués, les terminaux portuaires peuvent prédire et anticiper les besoins de placement des conteneurs. Cela permet de réduire les temps d'attente, d'optimiser l'utilisation de l'espace dans les yards de stockage, et d'améliorer l'efficacité globale des opérations de chargement et de déchargement. L'apprentissage automatique aide également à minimiser les manipulations inutiles en plaçant stratégiquement les conteneurs en fonction de leur destination et de leur priorité de sortie, contribuant ainsi à une gestion plus fluide et efficace des flux logistiques dans les ports.

## 1.7 Conclusion

En conclusion de ce chapitre, nous avons examiné les divers problèmes et limites liés au stockage des conteneurs, ainsi que les concepts mathématiques et informatiques utilisés dans ce domaine. Dans le chapitre suivant, nous approfondirons ces concepts et explorerons les travaux de recherche existants dans la littérature.

# Chapitre 2 :Techniques de détection d'emplacement et de prédiction dans le problème de stockage des conteneurs

## 2.1 Introduction

Dans ce chapitre, nous allons présenter la problématique que nous étudierons. Nous passerons en revue les différents articles présents dans la littérature qui ont traité de ce sujet, puis nous exposerons notre propre approche et nos objectifs de recherche.

## 2.2 Techniques utilisées pour déterminer l'emplacement des conteneurs

Déterminer l'emplacement des conteneurs représente un des défis mathématiques et informatiques complexes dont l'objectif est de trouver la meilleure solution parmi un ensemble de solutions possibles, tout en respectant certaines contraintes.déterminer l'emplacement de ces derniers, Ils se déclinent en plusieurs catégories :

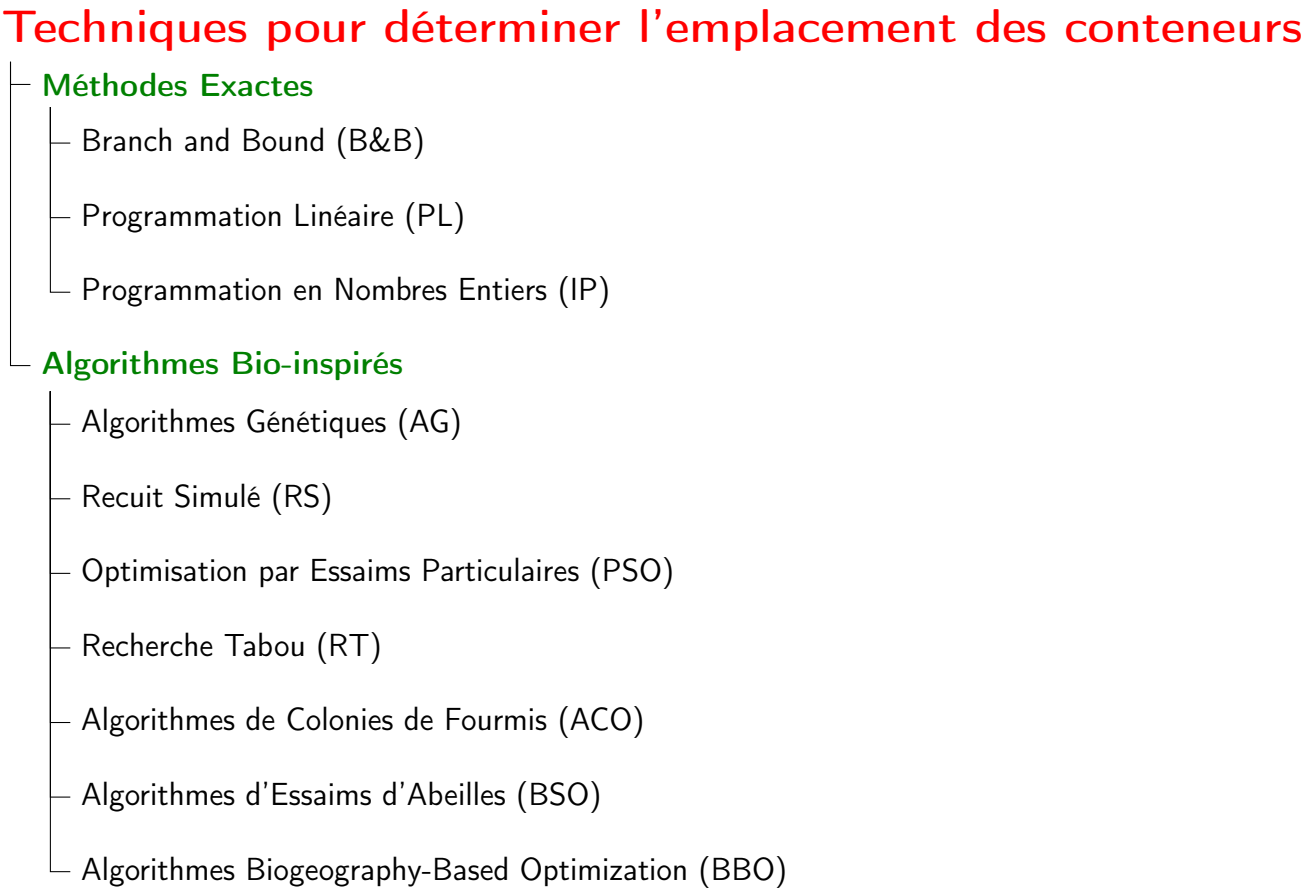



FIGURE 2.1 – Techniques pour déterminer l'emplacement des conteneurs

## 2.3 Méthodes Exactes

- **Branch and Bound (B&B)** : Divise le problème en sous-problèmes plus petits et utilise des bornes pour éliminer des sous-problèmes non prometteurs, voici un article qui parle de cette methode.[17]
- **Programmation Linéaire (PL)** : Optimise des fonctions linéaires sous contraintes linéaires,voici un livre qui parle de cette methode.[23]
- **Programmation en Nombres Entiers (IP)** : Optimise des fonctions avec des variables entières.[2]

## 2.4 Algorithmes bio-inspirés

Les algorithmes bio-inspirés sont des techniques de résolution de problèmes qui s'inspirent des principes observés dans la nature, comme le comportement des fourmis, des abeilles, ou même les processus évolutifs. Ces méthodes cherchent à reproduire des mécanismes biologiques pour trouver des solutions efficaces à des problèmes complexes, voici un livre qui résume ces méthodes[24] :

— **Algorithmes Génétiques (AG)** : Les Algorithmes Génétiques (AG) sont inspirés par la théorie de l'évolution de Darwin. Ils sont utilisés pour résoudre des problèmes d'optimisation en simulant des processus biologiques tels que la

sélection naturelle, le croisement et la mutation [26]. Initialement, une population de solutions potentielles est générée aléatoirement. Ensuite, les individus les plus adaptés, c'est-à-dire ceux qui correspondent le mieux à l'objectif défini, sont sélectionnés pour se reproduire et former une nouvelle génération. Ce processus se répète sur plusieurs générations, avec l'espoir que les solutions convergeront vers une solution optimale ou près de l'optimum global.

— **Recuit Simulé (RS)** : Le Recuit Simulé (RS) est inspiré par le processus de recuit utilisé en métallurgie pour améliorer la structure cristalline d'un matériau. En optimisation, le RS permet d'échapper aux optima locaux en acceptant initialement des solutions de qualité inférieure avec une certaine probabilité, qui diminue au fil du temps [14]. Cela permet à l'algorithme d'explorer un espace de recherche plus large à la recherche de la solution optimale. Le RS est particulièrement efficace pour les problèmes où la topologie de la fonction objectif est complexe et contient de multiples optima locaux.

— **Optimisation par Essaims Particulaires (PSO)** : Inspiré par le comportement des essaims, le PSO consiste en des particules qui se déplacent dans l'espace de recherche pour optimiser une fonction objectif. Chaque particule ajuste sa position en fonction de sa meilleure performance personnelle et de la meilleure performance globale trouvée par l'essaim, favorisant ainsi l'exploration et l'exploitation de l'espace de recherche [29].

— **Recherche Tabou (RT)** : Utilisant des stratégies de mémorisation, la Recherche Tabou évite de revisiter des solutions déjà explorées. Elle maintient une liste de mouvements "tabous" pour empêcher les cycles et promouvoir une exploration plus diversifiée de l'espace de recherche, améliorant ainsi la capacité à trouver des solutions de qualité [6].

— **Algorithmes de Colonies de Fourmis (ACO)** : Inspirés par le comportement des fourmis, les ACO sont utilisés pour résoudre des problèmes d'optimisation en imitant la recherche de nourriture par les fourmis. Les informations sur les chemins sont échangées entre les fourmis via des phéromones artificielles, permettant ainsi à la colonie de converger vers des solutions optimales en explorant et en exploitant efficacement l'espace de recherche [8].

— **Algorithmes d'Essaims d'Abeilles (BSO)** : Inspirés par le comportement des abeilles, les BSO sont utilisés pour optimiser des problèmes complexes en imitant la recherche de nourriture par les abeilles. Les abeilles coopèrent pour explorer de manière efficace l'espace de recherche, évaluant la qualité des sources de nourriture et communiquant les informations pour influencer les choix futures, favorisant ainsi la découverte de solutions optimales [1].

— **Algorithmes Biogeography-Based Optimization (BBO)** : BBO est inspiré par la biogéographie, qui étudie la distribution des espèces à travers les environnements géographiques. BBO modélise la migration des espèces entre différentes îles (ou habitats) en fonction de leur adaptabilité et de la qualité de leur habitat. Initialement, une population de solutions candidates est répartie sur ces îles. Au fil des générations, les solutions échangent des informations par migration, favorisant la propagation des solutions les plus adaptées [25].

## 2.5 Le problème de stockage des conteneurs dans la littérature

De nombreuses études liées au problème de stockage de conteneurs dans la littérature sur différents aspects des opérations des terminaux à conteneurs ont été publiées :

| Article | Auteur | Méthode | Avantage | Inconvénient |
|---|---|---|---|---|
| Solving the Container Problem using a Metaheuristic Genetic Algorithm | Marko et al. (2022)[10] | Algorithme Génétique | L'une des meilleures solutions par rapport aux solutions connues dans ce domaine de recherche. | Il n'est pas possible de vérifier la qualité de cette méthode pour de grands ensembles de tests (par exemple une baie de 6 niveaux et 6 piles). |
| Machine learning-driven algorithms for the container relocation problem | Canrong et al.(2020)[32] | - Algorithme de branch and bound itératif - Algorithme de recherche de faisceau - Apprentissage automatique (Random Forest (RF) et les règles d'association (AR)) | L'utilisation des méthodes de prévision, en l'occurrence celles basées sur l'apprentissage automatique, améliore significativement la qualité des bornes supérieures (UB). | Nécessite plus de ressources, ce qui se traduit par un temps d'exécution plus long. |
| A New Hybrid Salp Swarm-simulated Annealing Algorithm for the Container Stacking Problem | Mohamed El Wakil et al.(2020).[9] | L'Algorithme de la Nuée de Salpes (SSA) et l'Algorithme de Recuit Simulé (SA) pour une exploration et exploitation efficaces de l'espace des solutions. | - Hybridation efficace entre SSA et SA. - Adaptation au discret pour les problèmes d'optimisation. - Performance compétitive avec les solutions optimales connues. | - Peut nécessiter un ajustement fin des paramètres pour des problèmes spécifiques. - La performance peut varier en fonction de la nature du problème. |
| Optimisation du stockage de conteneurs dans un terminal portuaire(thése) | Nobar Antoine KASSABIAN (2022)[13] | Algorithme génetic,Recuit Simulé, Colonie d'Abeilles | Propose plusieurs algorithmes pour résoudre le problème de stockage de conteneurs | Nécessite trop de mémoire d'ordinateur et un temps de calcul coûteux |

TABLE 2.1 – Résumé des articles sur le problème du conteneur

**Marko et al. (2022)**[10] ont proposé une nouvelle méthode basée sur un algorithme génétique pour résoudre le problème restreint de relocalisation des conteneurs (CRP) dans la zone de stockage d'un terminal à conteneurs portuaire. Leur expérience a été menée sur des cas de test comprenant 20 tailles de baies différentes (avec divers nombres de niveaux et de piles), chacun composé de 40 instances de test distinctes. Ces instances de test sont les plus complexes existantes pour évaluer les modèles de résolution des CRP restreints, car elles contiennent toutes le nombre maximum possible de conteneurs (occupation maximale) en fonction de la taille de la baie du cas de test CRP particulier, la métrique utilisée pour évaluer l'algorithme est la

fonction de fitness et elle est de 266.04.
Les résultats expérimentaux prouvent que la méthode proposée détermine avec succès la relocalisation optimale des conteneurs bloquants afin de minimiser le nombre total de relocalisations dans la baie. Pour tous les cas de test, indépendamment de la taille de la baie, la méthode proposée s'est avérée être la meilleure ou la deuxième meilleure solution par rapport aux solutions bien connues d'autres auteurs dans ce domaine de recherche.
La seule limitation de cette méthode est qu'il n'est pas possible de vérifier la qualité des solutions obtenues pour les grands ensembles de tests (par exemple, une baie de stockage de 6 niveaux et 6 piles). Néanmoins, on peut supposer que la méthode trouve d'excellentes solutions puisqu'il a été prouvé qu'elle donne des résultats optimaux pour les petits ensembles de tests. Pour les grands ensembles de tests, cette méthode obtient les meilleurs ou les deuxièmes meilleurs résultats par rapport aux autres modèles ou méthodes pertinents qui résolvent le CRP.

**Canrong et al.(2020)**[32] ont proposé une nouvelle méthode de calcul des bornes supérieures pour un problème restreint de rèlocalisation de conteneurs dans les terminaux à conteneurs. Cette méthode intègre la méthode de prévision et l'apprentissage automatique pour faciliter la convergence. De plus, les bornes supérieures optimisées par apprentissage automatique sont injectées dans un algorithme exact de branchement et de bornes ainsi que dans une méthode de recherche par faisceau pour améliorer leurs performances respectives. Les expériences numériques montrent que la borne supérieure optimisée par apprentissage automatique, l'algorithme exact et l'heuristique affichent de meilleures performances que les méthodes de pointe rapportées dans la littérature basées sur les instances de test de référence.

**Mohamed El Wakil and al (2020).[9]** proposent une nouvelle version de l'algorithme des essaims de salpes (SSA) adaptée aux problèmes d'optimisation discrets. Ils hybrident ensuite cet algorithme pour créer l'algorithme hybride des essaims de salpes et du recuit simulé (SSSAA), qui combine les forces du SSA avec le recuit simulé (SA) afin de surmonter le piège des optima locaux. Le composant SA améliore l'exploitation en mettant à jour la salpe leader au sein de l'essaim. L'efficacité du SSSAA a été évaluée en le comparant aux meilleures solutions rapportées pour le problème de l'empilage des conteneurs (CSP). Étant donné que le CSP est un processus opérationnel nécessitant une exécution fréquente et rapide, le SSSAA a démontré une capacité impressionnante à trouver des solutions optimales pour la plupart des instances testées en un temps nettement inférieur à celui des méthodes de programmation en nombres entiers mixtes documentées dans la littérature. Les résultats computationnels révèlent que le SSSAA est un outil très rapide, efficace et performant pour résoudre le CSP.
**Nobar Antoine KASSABIAN (2022)** [13]a proposé plusieurs algorithmes pour résoudre le problème de stockage de conteneurs. La métrique principalement utilisée est le "Gap", exprimé en pourcentage. Cette métrique mesure l'écart entre la meilleure solution trouvée par l'algorithme et la meilleure solution connue pour chaque instance du problème. Voici un résumé de ces résultats :

— Les résultats obtenus pour l'algorithme AG montrent des écarts (Gaps) variant de 0.00% à 9.32% par rapport aux solutions optimales, avec un écart moyen de 3.93%.

— Pour le Recuit Simulé le Gap varie entre 0.00% et 5.66%, avec un écart moyen de 2.28%.
— Pour la Colonie d'Abeilles Le Gap varie de 0.00% à 5.66% pour l'algorithme BA et l'écart moyen est de 0.68% pour l'algorithme BA.

## 2.6 Techniques utilisées pour la prédiction dans les opérations portuaires

Dans les ports en général, les techniques de machine learning sont utilisées dans plusieurs domaines. Elles permettent d'analyser des données historiques et de prendre en compte divers facteurs influents comme les conditions météorologiques, les fluctuations économiques et les événements imprévus. En conséquence, les gestionnaires portuaires peuvent anticiper les besoins futurs, réduire les temps d'attente, minimiser les coûts et maximiser l'utilisation des infrastructures, assurant ainsi une gestion plus efficace et réactive des opérations portuaires notamment les opérations de stockage.

— **Régression Linéaire :** Technique utilisée pour modéliser la relation linéaire entre une variable dépendante continue et une ou plusieurs variables indépendantes. Elle est souvent utilisée pour prédire des valeurs numériques [30].
— **Régression Logistique :** Utilisée pour modéliser la relation entre une variable dépendante binaire (ou dichotomique) et des variables indépendantes. Elle est particulièrement adaptée à la classification binaire [16].
— **K plus proches voisins (KNN)** : Algorithme simple de classification et de régression qui fonctionne en trouvant les points de données les plus proches dans l'espace des caractéristiques pour prédire la valeur d'un nouveau point en utilisant une majorité de votes (pour la classification) ou une moyenne (pour la régression) [20].
— **GradientBoostingClassifier** : Le Gradient Boosting est un algorithme d'ensemble qui construit un modèle prédictif fort en combinant de manière itérative des modèles faibles, généralement des arbres de décision peu profonds. A chaque itération, un nouvel arbre est ajouté au modèle, entraîné pour corriger les erreurs de prédiction des arbres précédents. Le Gradient Boosting minimise une fonction de perte, comme la déviance, sur les données d'apprentissage. Cette approche itérative et additive permet de capturer des relations non linéaires complexes entre les variables et d'obtenir une grande précision prédictive. [5]
— **Forêts Aléatoires (FA)** :Cet algorithme d'ensemble repose sur la construction de multiples arbres de décision, chacun entraîné sur un sous-ensemble aléatoire des données d'entraînement et des variables prédictives. Chaque arbre de décision apprend à prédire la priorité du conteneur en fonction des caractéristiques qui lui sont présentées. La prédiction finale du Random Forest est obtenue en combinant les prédictions de tous les arbres par un vote majoritaire. Cette stratégie, appelée "bagging", permet de réduire la variance du modèle et de le rendre plus robuste face au surapprentissage. Le Random Forest est particulièrement adapté aux jeux de données contenant un grand nombre de variables et d'observations.[19][3].

— **XGBoost (XGB)** : Méthode d'ensemble basée sur le gradient boosting, utilisée pour des tâches de classification et de régression en construisant séquentiellement des modèles d'arbres de décision qui corrigent les erreurs des modèles précédents [27].
— **Réseaux de Neurones (ANN)** : Les réseaux neuronaux constituent un programme ou un modèle de machine learning qui prend des décisions d'une manière comparable au cerveau humain, en utilisant des processus qui reproduisent la façon dont les neurones biologiques fonctionnent de concert pour identifier des phénomènes, évaluer des options et arriver à des conclusions [22].

## 2.7 Prédiction dans les opérations portuaires dans la littérature

Dans la littérature, plusieurs articles ont traité de ces problèmes. Voici quelques exemples que nous avons trouvés :

| Article | Auteur | Méthode | Avantage | Inconvénient |
|---|---|---|---|---|
| Forecasting the Demand for Container Throughput Using a Mixed-Precision Neural Architecture Based on CNN–LSTM | Cheng-Hong Yang and Po-Yin Chang (2020)[31] | Combinaison de Convolutional Neural Network (CNN) avec Long Short-Term Memory (LSTM) | Amélioration significative des capacités prédictives pour les données de débit de conteneurs | Limitation aux données temporelles univariées, non valable pour des données plus complexes. |
| Prediction and Analysis of Container Terminal Logistics Arrival Time Based on Simulation Interactive Modeling : A Case Study of Ningbo Port | Ruoqi Wang and al.(2023)[28] | Arbre de decision, SVM, KNN, Adaptive Boosting, Random Forest | Améliore la précision et la fiabilité des prévisions des temps d'arrivée des conteneurs en utilisant des méthodes d'analyse de données et de modélisation par simulation. | La precision n'est toujours pas trés elevé(72%) |
| Container terminal daily gate in and gate out forecasting using machine learning methods | Jiahuan Jin and al.(2023)[12] | ARIMA, XGBoost | La méthode la plus efficace est l'ARIMA. | Les données utilisées sont légèrement obsolètes, ce qui pourrait limiter la pertinence et l'applicabilité des résultats dans le contexte actuel. |
| Development of Models Predicting Dwell Time of Import Containers in Port Container Terminals – An Artificial Neural Networks Application | Ioanna Kourounioti and al(2016)[15] | Réseaux de neurones | Utilise les réseaux de neurones qui n'ont jamais été utiliser avant cela. | Precision de 65%. |

TABLE 2.2 – Comparaison des méthodes de prédiction dans le problème de stockage des conteneurs

**Cheng-Hong Yang et Po-Yin Chang (2020**)[31] ont proposé une architecture neuronale à précision mixte pour la prévision du débit de conteneurs, une question cruciale dans le fonctionnement des ports. Leur architecture combinait les forces des CNN et des LSTM. Les opérations matricielles dans le modèle CNN-LSTM ont permis d'extraire des caractéristiques abstraites et de modéliser les dépendances

temporelles à partir des modèles CNN et LSTM. Cette approche a montré une extraction plus complète des caractéristiques pour l'ensemble des données, améliorant ainsi les performances de prévision par rapport aux modèles CNN ou LSTM seuls.

**Ruoqi Wang et al. (2023)**[28] ont proposé une étude utilisant une revue de la littérature, une enquête et une analyse de données pour explorer les facteurs influençant le temps d'arrivée des conteneurs. Leur revue de la littérature a identifié la congestion du trafic comme le principal facteur de retard. Cependant, les résultats de l'enquête n'ont montré aucune corrélation significative entre divers facteurs et les retards d'arrivée, malgré l'opinion des chauffeurs de camion attribuant principalement les retards à la congestion du trafic. Divers modèles de prédiction ont été testés, notamment le Random Forest (RF), Adaboost, K-Nearest Neighbors (KNN), Support Vector Machines (SVM) et Classification and Regression Trees (CART). Ces modèles, qu'ils utilisent des variables continues ou discrètes liées au trafic ou aux conditions météorologiques, ont atteint une précision limitée, avec un maximum de seulement 72%. Cela suggère que les modèles actuels peinent à prédire efficacement les temps d'arrivée des conteneurs.Malgré la congestion du trafic identifiée de manière constante comme un facteur majeur de retard dans la littérature et les enquêtes, l'étude souligne les limites des modèles de prédiction actuels pour intégrer et exploiter efficacement ces facteurs. De plus, les conditions météorologiques et les facteurs temporels spécifiques ont montré une corrélation minimale avec les temps d'arrivée, compliquant davantage la précision des prédictions.

**Jiahuan Jin et al. (2023)**[12] ont proposé un cadre de décomposition-ensemble pour prédire les entrées et sorties quotidiennes de conteneurs à un terminal à conteneurs. Leur méthode divise le problème quotidien en sous-problèmes de prévision basés sur les navires chaque jour, utilisant des caractéristiques temporelles spécifiques aux navires dans le modèle XGBoost. Les résultats individuels de chaque navire sont ensuite agrégés pour obtenir la prévision des entrées/sorties pour le jour suivant. Les performances des méthodes sont évaluées en utilisant des données réelles du terminal à conteneurs de Ningbo Zhoushan Port Beilun Second, et comparées à la méthode classique ARIMA. Les mesures d'erreur MAE, RMSE et MAPE sont calculées pour évaluer les performances. Les résultats montrent que la méthode proposée est plus efficace que l'ARIMA.

**Ioanna Kourounioti et al. (2016)**[15] ont proposé un cadre méthodologique pour permettre aux opérateurs portuaires de prédire le temps de séjour des conteneurs et, par conséquent, de prédire la charge de travail quotidienne liée à l'arrivée des camions pour le ramassage des conteneurs d'importation. Pour ce faire, des réseaux de neurones artificiels (RNA) ont été utilisés pour classifier les conteneurs en fonction des données acquises à partir du système d'exploitation du terminal (TOS).

## 2.8 Priorité des conteneurs dans les ports

Dans les ports en général, la priorité des conteneurs que nous définissons comme le conteneur le plus susceptible de partir en premier est déterminée de façon aléatoire. Cela signifie qu'il n'existe pas de règle fixe ou de système prédéfini pour décider quel

conteneur sera traité en premier, cette approche aléatoire peut avoir plusieurs conséquences et souligne l'importance d'établir une méthode structurée pour la gestion des priorités et ainsi déterminer le meilleur emplacement des conteneurs.

## 2.9 Conclusion

Dans ce chapitre, nous avons étudié divers articles portant sur le problème de l'emplacement des conteneurs et la prédiction associée aux opérations portuaires. Nous avons déduit que les méthodes les plus utilisées dans les ports pour détecter l'emplacement des conteneurs sont les méthodes bio-inspirées, contrairement aux méthodes exactes qui ne sont plus utilisées car elles sont moins efficaces. De plus, pour la prédiction dans le domaine des opérations portuaires, les techniques basées sur les réseaux de neurones sont moins intéressantes en raison de leur faible précision, contrairement aux méthodes classiques qui sont plus utilisées et plus efficaces.

Nous avons constaté que la priorité des conteneurs était prédéfinie et supposée connue pour chaque technique utilisée pour résoudre le problème. Définir manuellement la priorité d'un conteneur est un travail ardu et chronophage. C'est pourquoi, dans le chapitre suivant, nous proposons une nouvelle approche hybride pour la détection du meilleur emplacement du conteneur. Notre approche consiste à déterminer automatiquement la priorité de chaque conteneur. Ces priorités sont ensuite utilisées pour déterminer le meilleur emplacement de ces derniers.

# Chapitre 3 : Conception et réalisation

## 3.1 Introduction

Dans le port, la priorité des conteneurs est déterminée de façon aléatoire. Cela signifie qu'il n'existe pas de règle fixe ou de système prédéfini pour décider quel conteneur sera traité en premier, c'est pourquoi nous avons proposé une solution Hybride intelligente d'optimisation du placement des conteneurs dans un terminal portuaire. L'approche que nous proposons combine deux éléments clés : **Un modèle de prédiction de la priorité des conteneurs et un modèle de détermination du meilleur emplacement**. Dans ce qui suit nous détaillerons, notre approche, son implémentation et les résultats obtenus.

## 3.2 Approche développée

L'approche que nous proposons est représenté dans la figure 3.1 Le processus est composé de 3 modules, le premier pour le prétraitement des donnés, le second pour la prédiction des priorités et le dernier pour la détermination du meilleur emplacement des conteneurs.

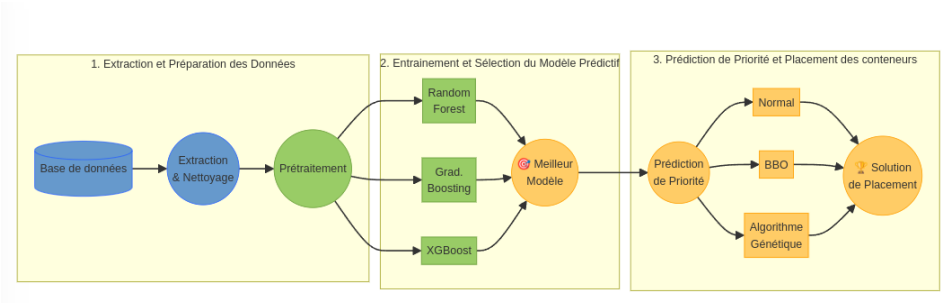



FIGURE 3.1 – Schéma de la solution d'optimisation

## 3.3 Prédiction de la Priorité des Conteneurs

L'organisation efficace du placement des conteneurs dans un terminal portuaire, visant à accélérer les opérations de déchargement, nécessite une compréhension approfondie des besoins spécifiques de chaque conteneur. Dans cette perspective, la prédiction de la priorité de traitement de chaque conteneur entrant est cruciale. Cette information de priorité sert de donnée d'entrée essentielle aux algorithmes de placement pour déterminer la meilleure position des conteneurs dans le terminal.

Ce modèle est développé et évalué dans la section 1.1, et exploite des données historiques de suivi des conteneurs et des algorithmes d'apprentissage automatique pour estimer la durée de séjour de chaque conteneur dans le terminal et lui assigner une priorité

Pour cela, nous avons proposé un modèle de prédiction capable d'estimer la durée de séjour d'un conteneur dans le terminal, et de lui assigner une priorité de 1 à 5 en fonction de cette durée. L'approche proposée est illustrée dans la figure 3.2. Elle repose sur l'exploitation de données historiques de suivi des conteneurs et sur l'utilisation d'algorithmes d'apprentissage automatique. Le modèle développé prend en compte diverses caractéristiques du conteneur, telles que le nom du navire, le transitaire, les lieux et dates de départ et d'arrivée, ainsi que les mois de dispatch, de déchargement et de livraison.

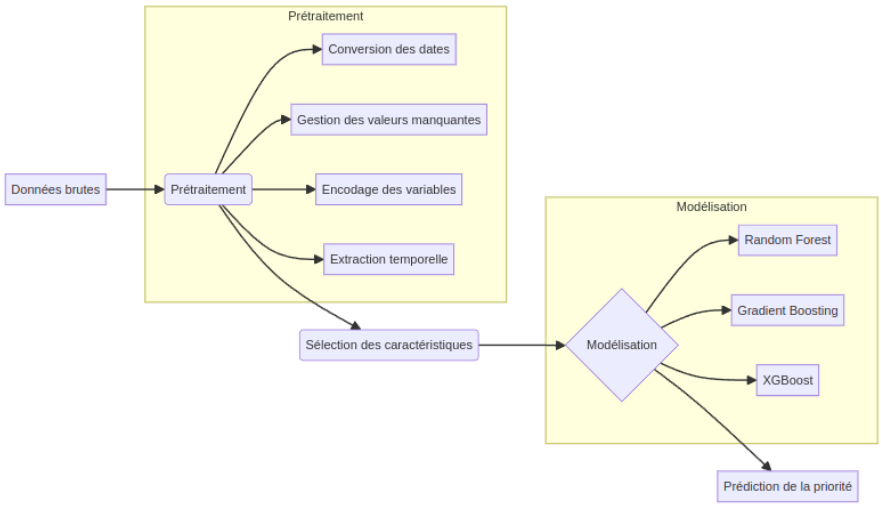



FIGURE 3.2 – Processus global de prédiction de la priorité des conteneurs.

Pour la prédiction de la priorité, nous évaluerons et comparerons trois algorithmes de classification : Random Forest, Gradient Boosting et XGBoost.

La performance des modèles de prédiction sera évaluée à l'aide de la métrique Mean Squared Error (MSE) et d'une matrice de confusion, permettant d'analyser la précision des prédictions pour chaque classe de priorité.

### 3.3.1 Choix des données et prétraitement

#### Informations sur les données

Avant de pouvoir construire notre modèle prédictif, il est essentiel de comprendre la nature des données mises à notre disposition. Le jeu de données utilisé, "Container Tracking Data.xlsx" [18], contient l'historique de suivi de conteneurs ayant transité par un terminal portuaire. Chaque enregistrement représente un conteneur unique, décrit par 16 variables qui offrent une vision complète de son parcours et de ses caractéristiques. Ces variables, résumées dans le Tableau 3.1, représentent des informations clés du conteneur, depuis son lieu d'origine jusqu'à sa destination finale, en passant par son séjour au terminal.

TABLE 3.1 – Description des variables du jeu de données

| **Variable** | **Description** |
|---|---|
| CONTAINER_NUMBER | Numéro unique attribué à chaque conteneur. |
| VESSEL_NAME | Nom du navire de transport. |
| FREIGHT_FORWARDER | Société de transport qui prend en charge le conteneur. |
| PLACE_OF_DISPATCH | Lieu de départ du conteneur. |
| PORT_OF_LOADING | Port où le conteneur est chargé sur le navire. |
| PORT_OF_DISCHARGE | Port où le conteneur est déchargé. |
| POST_PORT_OF_DISCHARGE | Destination après le port de déchargement. |
| PLACE_OF_DISPATCH_DATE | Date de départ du lieu de dispatch. |
| PORT_OF_LOADING_DATE | Date de chargement au port. |
| PORT_OF_DISCHARGE_DATE | Date de déchargement au port. |
| POST_PORT_OF_DISCHARGE_DATE | Date de départ après le port de déchargement. |
| PREDICTED_DELIVERED_DATE | Date de livraison prévue. |
| LAST_TRACKED_WITH_VESSEL | Dernière date de suivi avec le navire. |
| DELIVERED_FLAG | Indicateur de livraison (Oui ou Non). |
| DELIVERED_DATE | Date de livraison réelle. |
| Another NEW Predicated Delivery Date | Nouvelle date de livraison prédite (en cours de calcul). |

L'analyse de ces variables, combinée à des techniques de prétraitement adaptées, permettra au modèle de prédiction d'identifier les facteurs les plus déterminants pour estimer la durée de séjour d'un conteneur et lui assigner une priorité adéquate.

#### Conversion des dates en format datetime

Les variables relatives aux dates sont initialement importées en tant que chaînes de caractères. Afin de pouvoir les manipuler correctement et extraire des informations pertinentes pour la prédiction (ex : durée entre deux événements), il est nécessaire de les convertir au format datetime. Ce format permet de réaliser des opérations arithmétiques sur les dates, comme le calcul de la durée entre deux événements, ce qui est crucial pour déterminer la durée de séjour d'un conteneur dans le terminal.

#### Calcul de la durée de séjour du conteneur

Une information cruciale pour déterminer la priorité d'un conteneur est sa durée de séjour dans le port. Cette durée, qui représente le temps écoulé entre le déchargement du conteneur du navire et sa livraison effective à sa destination

finale, est calculée pour chaque conteneur en soustrayant la date de déchargement ('PORT_OF_DISCHARGE_DATE') de la date de livraison réelle ('DELIVERED_DATE'). La durée de séjour, exprimée en nombre de jours, est ensuite ajoutée au jeu de données sous forme d'une nouvelle variable nommée 'DURATION_OF_STAY'. Cette variable sera ensuite utilisée pour la création des classes de priorité.

### Gestion des valeurs manquantes

Les données brutes extraites du système d'information du terminal portuaire comportent inévitablement des valeurs manquantes pour certaines variables et certains conteneurs. Ces valeurs manquantes peuvent être dues à des erreurs de saisie, à des problèmes de transmission des données, ou à des informations incomplètes. Il est crucial de traiter ces valeurs manquantes avant d'entraîner le modèle de prédiction, car elles peuvent biaiser les résultats et affecter la performance du modèle.

Dans le cadre de notre étude, nous avons opté pour une stratégie simple et efficace : la suppression des lignes contenant des valeurs manquantes pour les variables clés, à savoir la date de déchargement du conteneur ('PORT_OF_DISCHARGE_DATE') et la date de livraison réelle ('DELIVERED_DATE'). Cette stratégie est justifiée par le fait que ces deux dates sont essentielles pour le calcul de la durée de séjour du conteneur, qui est la base de notre prédiction de priorité. Supprimer les lignes avec des valeurs manquantes pour ces variables nous permet de garantir la fiabilité du calcul de la durée de séjour et d'éviter d'introduire des biais dans le modèle.

D'autres stratégies, telles que l'imputation par la moyenne, la médiane, ou des algorithmes plus sophistiqués, auraient pu être envisagées. Cependant, compte tenu de la nature des données manquantes et de l'importance des variables concernées, nous avons estimé que la suppression des lignes était la solution la plus appropriée pour garantir la qualité et la fiabilité du modèle de prédiction.

### Transformation de la durée de séjour en classes de priorité

La durée de séjour du conteneur, calculée précédemment et stockée dans la variable `DURATION_OF_STAY`, est une variable numérique continue. Cependant, notre objectif est de prédire la priorité du conteneur, qui est une variable catégorielle discrète. Afin de simplifier le problème et de le rendre compatible avec un modèle de classification, nous transformons la durée de séjour en classes de priorité. Cinq classes de priorité sont définies, allant de 1 (priorité maximale) à 5 (priorité minimale), correspondant à des intervalles de durée de séjour croissants. La fonction `assign_priority` dans le code source implémente cette transformation en appliquant les règles suivantes :

- **Priorité 1 :** Durée de séjour inférieure ou égale à 3 jours.
- **Priorité 2 :** Durée de séjour entre 4 et 7 jours.
- **Priorité 3 :** Durée de séjour entre 8 et 14 jours.
- **Priorité 4 :** Durée de séjour entre 15 et 21 jours.
- **Priorité 5 :** Durée de séjour supérieure à 21 jours.

Ce découpage en classes de priorité permet de regrouper les conteneurs ayant des durées de séjour similaires et de simplifier la tâche de prédiction pour le modèle. La figure 3.3 illustre la distribution des conteneurs dans chaque classe de priorité.

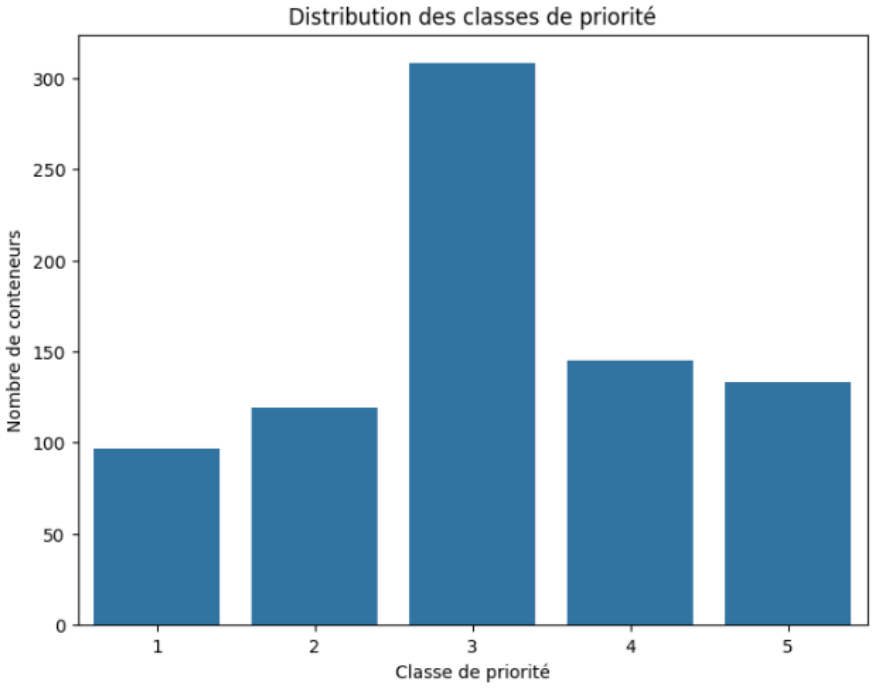



FIGURE 3.3 – Distribution des classes de priorité des conteneurs.

Comme le montre la figure 3.3, la classe de priorité 3 est la plus représentée, ce qui indique qu'une majorité de conteneurs ont une durée de séjour dans le terminal comprise entre 8 et 14 jours. La classe 2 et la classe 4 suivent avec un nombre légèrement inférieur de conteneurs, tandis que les classes 1 (priorité maximale) et 5 (priorité minimale) sont les moins représentées.

### Encodage des variables catégorielles

Certaines variables du jeu de données sont catégorielles, c'est-à-dire qu'elles prennent des valeurs discrètes représentant des catégories (ex : nom du navire, port de chargement). Ces variables, bien que riches en informations, ne peuvent pas être directement utilisées par les algorithmes d'apprentissage automatique qui nécessitent des données numériques.

Pour pallier ce problème, nous appliquons la technique d'"encodage par étiquettes" (Label Encoding). Cette technique consiste à transformer chaque valeur unique d'une variable catégorielle en un nombre entier unique. Par exemple, si la variable `VESSEL_NAME` contient les valeurs "Navire A", "Navire B" et "Navire C", ces valeurs seront encodées respectivement en 0, 1 et 2.

Le code Python utilise la classe 'LabelEncoder' de la librairie scikit-learn pour réaliser cet encodage. La boucle 'for' parcourt la liste 'categorical_columns' qui contient les noms des variables catégorielles à encoder : `VESSEL_NAME`, `FREIGHT_FORWARDER`, `PLACE_OF_DISPATCH`, `PORT_OF_LOADING`, `PORT_OF_DISCHARGE`,et `POST_PORT_OF_DISCHARGE`. Pour chaque variable, un objet 'LabelEncoder' est créé et utilisé pour transformer les valeurs textuelles en valeurs numériques.

L'encodage par étiquettes permet de conserver l'information contenue dans les variables catégorielles tout en les rendant compatibles avec les algorithmes d'apprentissage automatique.

### Extraction de caractéristiques temporelles

Les variables temporelles, telles que les dates de dispatch, de déchargement et de livraison, contiennent des informations potentiellement précieuses pour la prédiction de la priorité des conteneurs. Cependant, utiliser directement ces dates brutes comme caractéristiques d'entrée pour le modèle d'apprentissage automatique peut s'avérer inefficace. En effet, les algorithmes d'apprentissage automatique ont souvent du mal à interpréter directement les dates, car elles représentent des valeurs numériques continues qui n'ont pas de signification intrinsèque pour le modèle.

Pour exploiter au mieux l'information contenue dans les dates, nous extrayons des caractéristiques temporelles plus pertinentes. Dans notre code, nous extrayons le mois de l'année pour chaque date clé : le mois de dispatch ('DISPATCH_MONTH'), le mois de déchargement ('DISCHARGE_MONTH') et le mois de livraison ('DELIVERED_MONTH'). Ces caractéristiques, représentées par des nombres entiers de 1 à 12, permettent de capturer la saisonnalité potentielle du trafic portuaire et son impact sur la durée de séjour des conteneurs. Par exemple, certains mois de l'année peuvent être plus chargés que d'autres, entraînant des délais de traitement plus longs.

L'extraction de caractéristiques temporelles pertinentes permet d'améliorer la performance du modèle de prédiction en fournissant des informations plus significatives et plus faciles à interpréter pour l'algorithme d'apprentissage automatique.

### Sélection des caractéristiques pertinentes pour la prédiction

L'étape de sélection des caractéristiques est cruciale pour construire un modèle de prédiction performant et efficace. Afin d'identifier les variables les plus pertinentes, nous avons calculé la matrice de corrélation entre les variables pré-traitées, incluant les caractéristiques temporelles extraites, et la variable cible *PRIORITY*. La Figure 3.4 présente la matrice de corrélation générée par notre code Python. Cette matrice permet de visualiser les relations linéaires entre les variables et d'identifier celles qui sont les plus fortement corrélées avec la priorité du conteneur.

L'analyse de cette matrice, combinée à des connaissances métier, nous a permis de sélectionner les variables les plus informatives pour la prédiction de la priorité des conteneurs. Nous remarquons ue la matrice inclut maintenant les caractéristiques temporelles extraites (DISPATCH_MONTH, DISCHARGE_MONTH, DELIVERED_MONTH).

Dans notre code, nous avons retenu les variables suivantes :

*__VESSEL_NAME__ : Le nom du navire peut être un indicateur de la fiabilité du transporteur et de la fréquence des retards. Bien que la corrélation linéaire avec la variable *PRIORITY* soit faible, nous avons choisi de conserver cette variable car elle peut potentiellement apporter des informations utiles de manière non linéaire. De plus, le nom du navire est souvent associé à des caractéristiques spécifiques (taille, type de cargaison, routes habituelles) qui peuvent influencer la durée de séjour d'un conteneur.

* **FREIGHT_FORWARDER** : Le transitaire peut influencer les délais de livraison en fonction de ses processus logistiques et de son efficacité.

* **PLACE_OF_DISPATCH**, **PORT_OF_LOADING**, **PORT_OF_DISCHARGE**, **POST_PORT_OF_DISCHARGE** : Les lieux de départ, de chargement, de

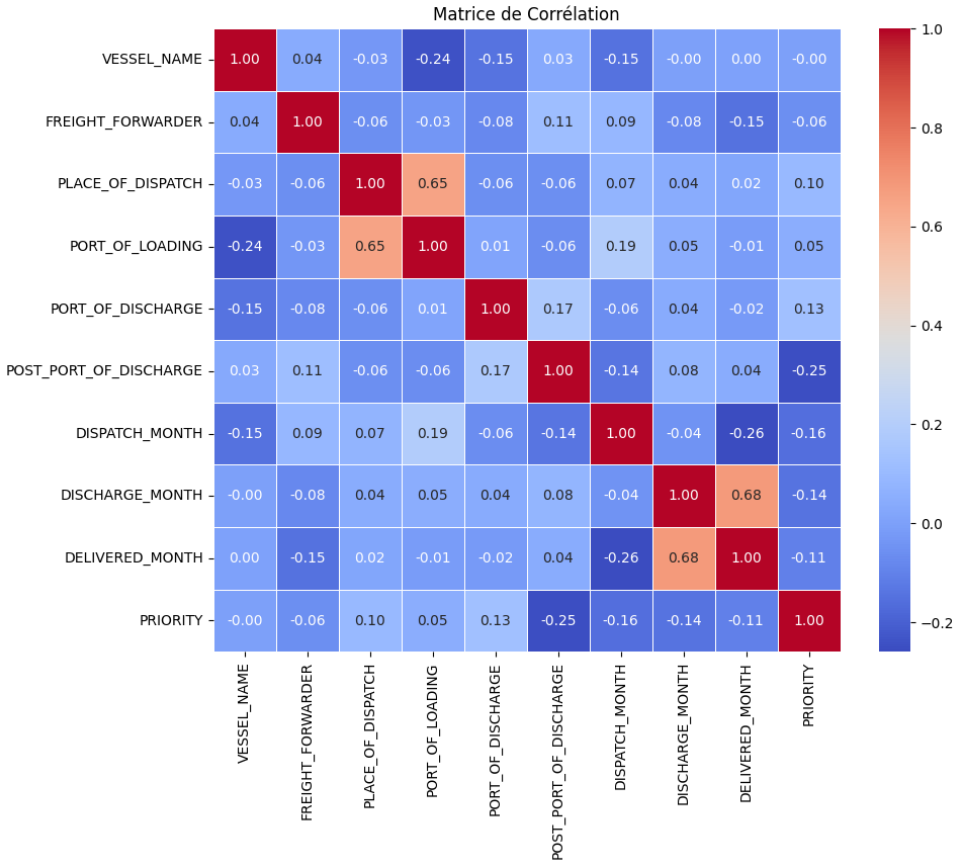



FIGURE 3.4 – Matrice de corrélation entre les variables

déchargement et de destination finale du conteneur peuvent influencer la durée de séjour en fonction des distances parcourues et des processus douaniers.
* **DISPATCH_MONTH, DISCHARGE_MONTH, DELIVERED_MONTH** : Les mois de l'année pour chaque date clé permettent de capturer la saisonnalité du trafic portuaire et son impact sur les délais de traitement.

Ces caractéristiques sont ensuite utilisées pour construire la matrice X qui servira à entraîner le modèle de prédiction.

### Division en ensembles d'entraînement et de test

Une fois les caractéristiques pertinentes sélectionnées, il est nécessaire de diviser le jeu de données en deux ensembles distincts : un ensemble d'entraînement et un ensemble de test. Cette étape est cruciale pour évaluer la performance du modèle de prédiction de manière fiable et objective.

L'ensemble d'entraînement est utilisé pour construire le modèle. L'algorithme d'apprentissage automatique apprend à partir de ces données en identifiant les relations entre les caractéristiques sélectionnées et la variable cible *PRIORITY*. Dans notre code, nous utilisons 80% des données pour l'entraînement, ce qui permet au modèle d'apprendre à partir d'un échantillon suffisamment large pour capturer les tendances et les patterns du jeu de données.

L'ensemble de test, quant à lui, est utilisé pour évaluer les performances du modèle sur des données qu'il n'a jamais vues auparavant. En comparant les prédictions

du modèle sur l'ensemble de test avec les valeurs réelles de la variable cible, nous pouvons mesurer sa capacité à généraliser ses apprentissages à de nouvelles données. Dans notre code, nous réservons 20% des données pour le test, ce qui permet d'obtenir une estimation fiable de la performance du modèle en situation réelle.

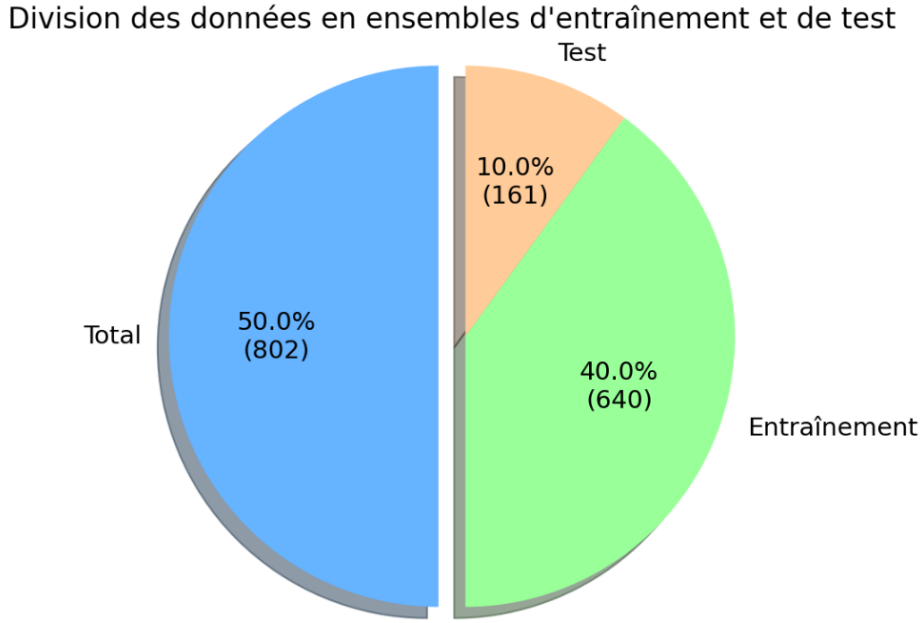



FIGURE 3.5 – Division du jeu de données en ensembles d'entraînement et de test.

La division du jeu de données en ensembles d'entraînement et de test est réalisée dans notre code Python à l'aide de la fonction `train_test_split` de la librairie scikit-learn. Cette fonction permet de diviser les données de manière aléatoire, tout en garantissant un ratio 80/20 entre l'entraînement et le test. Le paramètre `random_state` est utilisé pour fixer la graine du générateur de nombres aléatoires, ce qui permet de reproduire les mêmes partitions des données à chaque exécution du code.

Cette division rigoureuse en ensembles d'entraînement et de test est essentielle pour garantir la fiabilité de l'évaluation du modèle et éviter le surapprentissage, c'est-à-dire un modèle qui serait trop spécialisé sur les données d'entraînement et qui aurait des difficultés à généraliser à de nouvelles données.

### Normalisation des caractéristiques

Après la division du jeu de données en ensembles d'entraînement et de test, nous réalisons une étape de normalisation des caractéristiques. Cette étape est importante car les variables du jeu de données peuvent avoir des échelles et des unités différentes, ce qui peut poser problème lors de l'utilisation de certains algorithmes d'apprentissage automatique. Par exemple, la variable `VESSEL_NAME`, après encodage, prend des valeurs entières allant de 0 à N (N étant le nombre de navires uniques), tandis que la variable `DISPATCH_MONTH` prend des valeurs entières de 1 à 12. Ces différences d'échelle peuvent entraîner un biais dans l'apprentissage du modèle, certains algorithmes étant sensibles aux variations d'amplitude des variables.

La normalisation des caractéristiques permet de mettre toutes les variables sur une même échelle, sans modifier leur distribution. Dans notre code, nous utilisons la technique de standardisation, implémentée par la classe `StandardScaler` de la librairie scikit-learn. Cette technique consiste à centrer et réduire chaque variable, de sorte que sa moyenne soit égale à 0 et son écart type égal à 1.

La standardisation est appliquée à la fois à l'ensemble d'entraînement et à l'ensemble de test. La fonction `fit_transform` est utilisée pour ajuster le `StandardScaler` sur les données d'entraînement et ensuite transformer ces données. La fonction `transform` est utilisée pour appliquer la même transformation à l'ensemble de test, en utilisant les paramètres appris sur l'ensemble d'entraînement.

En normalisant les caractéristiques, nous garantissons que toutes les variables ont une influence équivalente sur l'apprentissage du modèle, ce qui permet d'améliorer sa performance et sa robustesse.

### 3.3.2 Modélisation Prédictive

La modélisation prédictive de la priorité des conteneurs représente le cœur de notre solution d'optimisation des opérations portuaires. Cette phase est cruciale car elle va permettre d'alimenter l'algorithme d'optimisation avec des informations fiables sur l'ordre de traitement optimal des conteneurs.

#### a) Choix des modèles de classification

Le problème que nous traitons ici est un problème de classification multiclasse : il s'agit d'assigner à chaque conteneur entrant une classe de priorité parmi un nombre fini de classes prédéfinies (5 dans notre cas). De plus, les relations entre les variables prédictives (caractéristiques du conteneur) et la variable cible (priorité) peuvent être complexes et non linéaires. Face à ces exigences, nous avons sélectionné trois algorithmes d'apprentissage automatique reconnus pour leur performance et leur capacité à gérer des problèmes de classification multiclasse avec des relations non linéaires :

(a) **RandomForestClassifier**
(b) **GradientBoostingClassifier**
(c) **XGBoostClassifier**

En choisissant ces trois algorithmes, nous augmentons ainsi nos chances de trouver un modèle de prédiction performant et adapté à notre problématique.

#### b) Recherche d'hyperparamètres optimaux

Chaque modèle de classification possède un ensemble d'hyperparamètres qui contrôlent son fonctionnement et influencent sa performance finale. Il est donc crucial de trouver les valeurs optimales de ces hyperparamètres pour chaque modèle afin d'optimiser ses performances sur notre problème spécifique.

Pour ce faire, nous utilisons une approche systématique basée sur une recherche par grille ("Grid Search") combinée à une validation croisée à K plis (K=3 dans notre cas). La recherche par grille consiste à définir une grille de valeurs possibles pour

chaque hyperparamètre et à tester toutes les combinaisons possibles de ces valeurs. La validation croisée à K plis permet d'évaluer la performance de chaque combinaison d'hyperparamètres sur différentes partitions des données d'entraînement, ce qui permet d'obtenir une estimation plus robuste de la performance du modèle et de limiter le risque de surapprentissage.

Les principaux hyperparamètres que nous optimisons pour chaque modèle sont :

- **Nombre d'estimateurs (n_estimators)** : Ce paramètre contrôle le nombre d'arbres de décision dans l'ensemble. Augmenter le nombre d'estimateurs peut améliorer la performance du modèle, mais aussi augmenter le temps de calcul.
- **Profondeur maximale des arbres (max_depth)** : Ce paramètre limite la profondeur maximale de chaque arbre de décision dans l'ensemble. Une profondeur plus importante permet de capturer des relations plus complexes, mais peut aussi conduire à un surapprentissage.
- **Taux d'apprentissage (learning_rate)** : Ce paramètre contrôle l'impact de chaque arbre sur la mise à jour du modèle dans le cas du Gradient Boosting et du XGBoost. Un taux d'apprentissage plus faible peut améliorer la généralisation du modèle, mais nécessite plus d'itérations pour converger.

La combinaison d'hyperparamètres qui donne la meilleure performance moyenne sur les K plis de validation croisée sera retenue pour chaque modèle. Les meilleurs hyperparamètres trouvés pour chaque modèle sont présentés dans le Tableau 3.2.

TABLE 3.2 – Meilleurs hyperparamètres pour chaque modèle.

| **Modèle** | **n_estimators** | **max_depth** | **learning_rate** |
|---|---|---|---|
| RandomForest | 300 | 5 | - |
| GradientBoosting | 100 | 4 | 0.1 |
| XGBoost | 100 | 3 | 0.1 |

### c) Métrique d'évaluation : l'erreur quadratique moyenne (MSE)

Pour évaluer objectivement les performances de nos modèles de classification et sélectionner le plus performant, nous avons besoin d'une métrique d'évaluation robuste. Dans ce travail, nous utilisons l'erreur quadratique moyenne (Mean Squared Error, MSE) [7, 11, 21, 4].

**Définition et calcul de la MSE** La MSE, traditionnellement utilisée pour évaluer les modèles de régression, peut également s'appliquer à la classification multiclasse avec des classes ordonnées. Dans notre cas, la MSE quantifie la précision de la prédiction des priorités des conteneurs, qui sont classées de 1 à 5.

Concrètement, la MSE calcule la moyenne des carrés des différences entre les priorités prédites par le modèle et les priorités réelles. La formule mathématique de la MSE est :

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

où :

**n** est le nombre d'observations (conteneurs),
$\mathbf{y_i}$ est la vraie priorité du conteneur **i**,
$\hat{\mathbf{y}}_\mathbf{i}$ est la priorité prédite pour le conteneur **i**.

**Illustration du calcul de la MSE** Pour clarifier le calcul de la MSE, considérons un exemple avec 5 conteneurs et leurs priorités réelles et prédites, comme présenté dans le tableau 3.3.

TABLE 3.3 – Exemple de calcul de la MSE

| Conteneur | Priorité réelle | Priorité prédite | Différence | Différence$^2$ |
|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 0 |
| 2 | 2 | 3 | -1 | 1 |
| 3 | 3 | 2 | 1 | 1 |
| 4 | 4 | 5 | -1 | 1 |
| 5 | 5 | 4 | 1 | 1 |
| Somme des carrés des différences : | | | | **4** |

La MSE se calcule en divisant la somme des carrés des différences (4) par le nombre de conteneurs (5), soit : 4 / 5 = **0.8**.

**Interprétation de la MSE dans notre contexte** Plus la MSE est faible, plus les prédictions du modèle sont précises. Une MSE de 0 indique une prédiction parfaite. Dans notre cas, les valeurs possibles de la MSE varient de **0 à 16**, la valeur maximale étant atteinte lorsque le modèle prédit systématiquement la classe la plus éloignée de la classe réelle.
Une MSE faible est cruciale pour l'optimisation du placement des conteneurs car elle garantit un classement précis par ordre de priorité. Des erreurs importantes de prédiction pourraient induire un mauvais classement, affectant l'efficacité des opérations portuaires.

### d) Comparaison des performances des modèles de classification

Après avoir entraîné et évalué trois modèles de classification (Random Forest, Gradient Boosting et XGBoost) sur notre jeu de données, il est crucial de comparer leurs performances afin de sélectionner le modèle le plus adapté à la prédiction de la priorité des conteneurs. Pour une comparaison objective, nous utilisons la métrique d'évaluation MSE (Mean Squared Error) qui mesure l'écart quadratique moyen entre les priorités prédites et les priorités réelles. Plus la MSE est faible, meilleure est la performance du modèle. Enfin, nous analysons les matrices de confusion de chaque modèle pour visualiser leur capacité à prédire correctement les différentes classes de priorité.

### e) Résultats et analyse

Le tableau 3.4 présente une comparaison des MSE obtenues pour chaque modèle. XGBoost se distingue avec la MSE la plus faible (0.745), indiquant une meilleure précision prédictive que Random Forest et Gradient Boosting.

TABLE 3.4 – Comparaison des performances des modèles

| **Modèle** | **MSE** |
|---|---|
| Random Forest | 0.956 |
| Gradient Boosting | 0.919 |
| XGBoost | **0.745** |

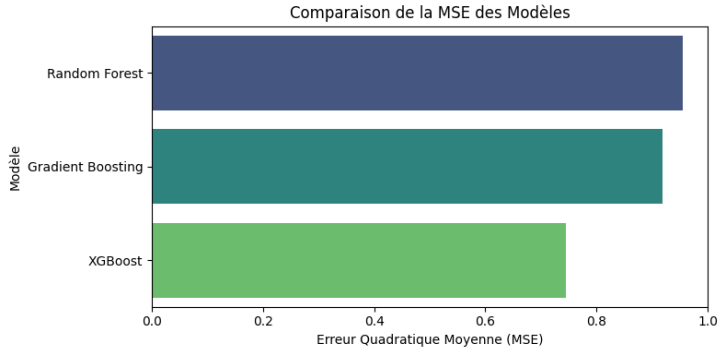



FIGURE 3.6 – Comparaison visuelle des MSE des trois modèles

Les matrices de confusion, illustrées dans les Figures 3.7 à 3.9, permettent d'analyser plus en détail la capacité de chaque modèle à prédire correctement les différentes classes de priorité. Une concentration élevée de prédictions correctes sur la diagonale de la matrice indique une meilleure performance.

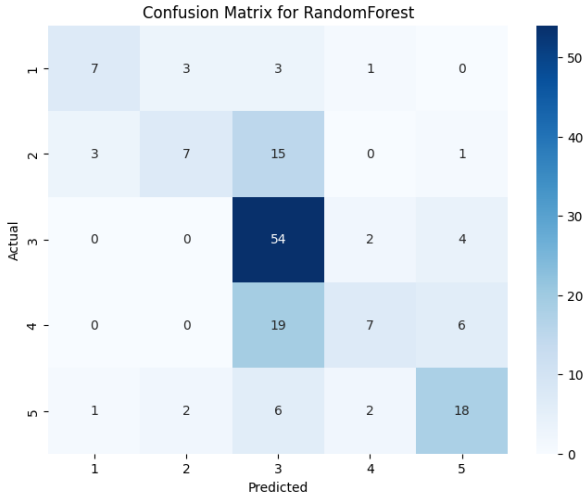



FIGURE 3.7 – Random Forest

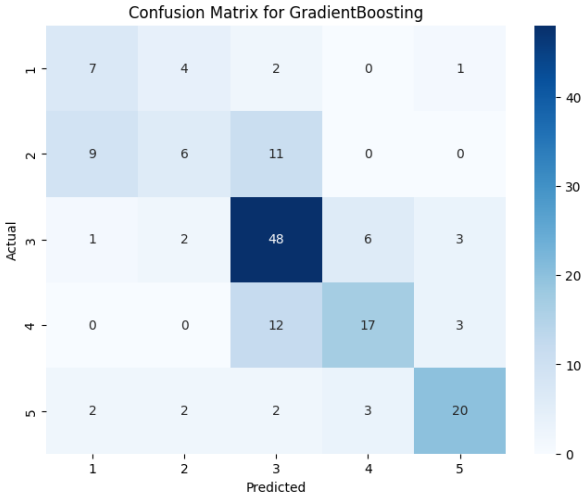



FIGURE 3.8 – Gradient Boosting

L'analyse des matrices de confusion confirme la supériorité de XGBoost. Visuellement, XGBoost présente une meilleure concentration de prédictions correctes sur la diagonale de sa matrice de confusion que les autres modèles.

**Conclusion :** XGBoost se révèle être le modèle le plus performant pour la prédiction de la priorité des conteneurs, avec la MSE la plus faible et une meilleure distribution des prédictions correctes sur sa matrice de confusion. Sa capacité à gérer la complexité des données et à prévenir le surapprentissage le positionne comme le choix optimal pour notre solution.

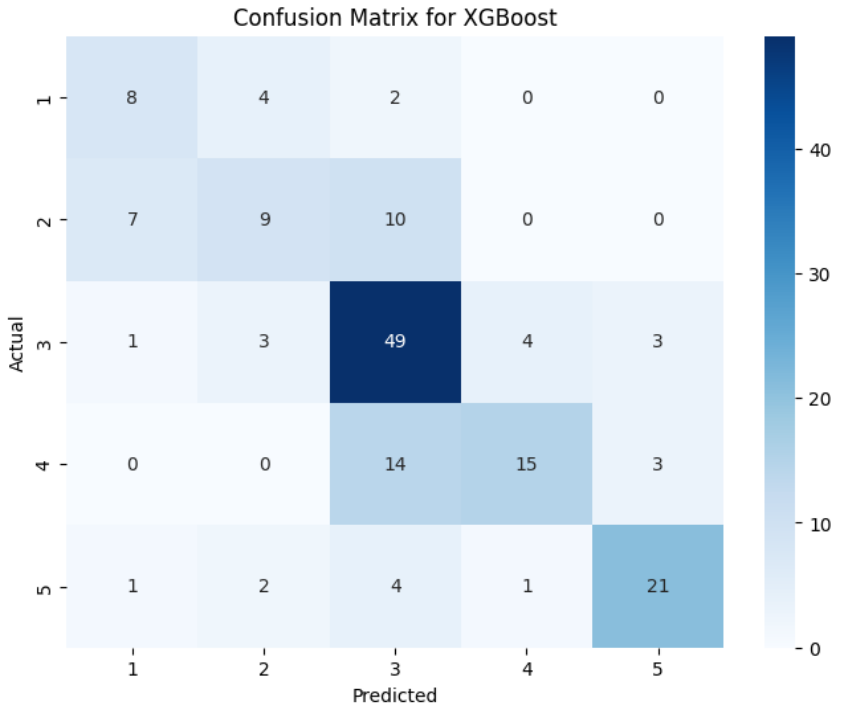



FIGURE 3.9 – XGBoost

## 3.4 Intégration de la Prédiction de Priorité dans l'Optimisation du Placement des Conteneurs

La prédiction de la priorité des conteneurs, réalisée dans la première partie de ce mémoire, fournit une information essentielle pour optimiser le placement des conteneurs dans le terminal. En connaissant l'urgence de chaque conteneur, il est possible de les placer de manière stratégique pour minimiser le temps de manutention global et faciliter l'accès aux conteneurs prioritaires.

Ce chapitre explore l'intégration de la prédiction de priorité dans trois algorithmes : "Normal" (recherche exhaustive), Biogeography-Based Optimization (BBO) et Algorithme Génétique (AG). Ces algorithmes sont adaptés au problème spécifique du placement des conteneurs, en tenant compte des contraintes du terminal (capacité des rangées, accessibilité) et en utilisant la prédiction de priorité pour guider le placement des conteneurs.

La performance de chaque algorithme est évaluée en fonction de la fonction *Fitness*, implémentée dans le code source, qui calcule le temps de manutention total de la grue de cour en tenant compte des mouvements horizontaux et verticaux, et en pénalisant les placements qui entravent l'accès aux conteneurs prioritaires. L'objectif est de comparer les algorithmes en termes de temps de manutention et de temps de calcul, afin de déterminer la solution la plus performante et la plus adaptée à un contexte portuaire réel.

La figure 3.10 illustre le processus. La prédiction de priorité alimente les trois algorithmes, qui convergent tous vers la détermination du meilleur emplacement des conteneurs.

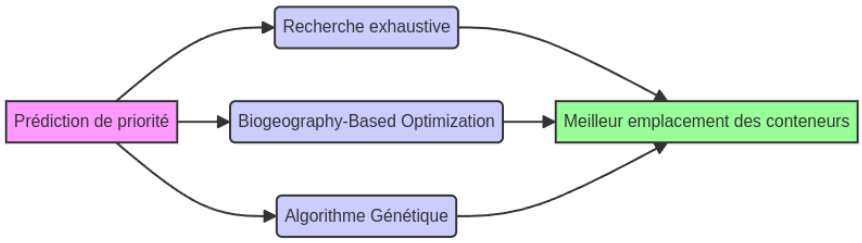



FIGURE 3.10 – Schéma du processus de placement des conteneurs

### 3.4.1 Métrique d'évaluation de la performance

— **Définition précise de la fonction objectif utilisée pour mesurer la qualité des solutions :** Pour évaluer objectivement la performance des algorithmes d'optimisation du placement des conteneurs, nous utilisons la fonction *Fitness* définie dans le code source. Cette fonction calcule un score pour chaque solution potentielle, représentant le temps total de manutention nécessaire pour placer tous les conteneurs entrants selon la configuration proposée par la solution.

```python
def Fitness(solution):
    """Calcule le fitness d'une solution d'emplacement des conteneurs."""
    value = 0
    dechargement_cost = 10  # Temps pour décharger un conteneur
    rows = []

    for zero, facteur in solution:
        # Coût de déplacement horizontal (aller-retour)
        value += 2 * MoveTimes[zero]

        # Coût de déplacement vertical
        niveau = EmptyCells[zero] - rows.count(zero)
        value += ShiftTimes[niveau] / facteur

        # Coût de déchargement
        conteneurs_dessus = rows.count(zero)
        value += conteneurs_dessus * dechargement_cost

        rows.append(zero)

    return value
```

FIGURE 3.11 – Illustration simplifiée de la fonction *Fitness*

— **Explication des critères pris en compte dans la fonction objectif (temps de manutention, nombre de déplacements, utilisation de l'espace, etc.) :** La fonction *Fitness* prend en compte les éléments suivants pour estimer le temps de manutention :

- **Temps de déplacement horizontal :** La fonction calcule le temps nécessaire à la grue de cour pour se déplacer horizontalement entre les rangées du bloc de stockage. Ce temps est proportionnel à la distance entre la position actuelle de la grue et la rangée où le conteneur doit être placé. Le code utilise le dictionnaire *MoveTimes* pour déterminer le temps de déplacement horizontal en fonction de la rangée.
- **Temps de déplacement vertical :** La fonction prend également en compte le temps nécessaire à la grue pour se déplacer verticalement jusqu'au niveau (tier) où le conteneur doit être placé dans la rangée. Ce temps est proportionnel à la hauteur du niveau, le niveau 1 étant le plus haut et le niveau 5 le plus bas. Le dictionnaire *ShiftTimes* permet de déterminer le temps de déplacement vertical en fonction du niveau.
- **Priorité du conteneur :** La fonction *Fitness* intègre la priorité des conteneurs en pondérant le temps de déplacement vertical. Plus la priorité du conteneur est élevée (1 étant la plus haute), plus le temps de déplacement vertical est pénalisé. Ce mécanisme encourage les algorithmes d'optimisation à placer les conteneurs prioritaires dans les niveaux supérieurs des rangées, facilitant ainsi leur accès et leur retrait ultérieur.
- **Pénalité d'empilement :** La fonction *Fitness* applique une pénalité pour chaque conteneur empilé au-dessus d'un conteneur de même priorité. Cette pénalité est justifiée par le fait qu'empiler des conteneurs de même priorité rend plus difficile l'accès au conteneur du dessous si celui-ci doit être retiré avant celui du dessus.
- **Coût de déchargement :** La fonction *Fitness* ajoute un coût de déchargement pour chaque conteneur au-dessus du conteneur cible. Ce coût représente le temps supplémentaire nécessaire pour déplacer les conteneurs empilés afin d'accéder au conteneur cible.

**Note :** Bien que la fonction *Fitness* ne prenne pas explicitement en compte l'utilisation de l'espace de stockage comme critère d'optimisation, elle favorise indirectement sa minimisation en encourageant le placement compact des conteneurs afin de réduire les temps de déplacement de la grue. L'objectif principal reste la minimisation du temps de manutention en priorisant l'accès rapide aux conteneurs urgents.

- **Justification du choix de cette métrique par rapport aux objectifs de l'optimisation :** Le choix de la fonction *Fitness* comme métrique d'évaluation est justifié par les objectifs suivants :

  **Minimisation du temps de manutention** et **Prise en compte de la priorité des conteneurs :** La fonction *Fitness* est directement corrélée au temps total nécessaire à la grue pour placer tous les conteneurs entrants. En minimisant cette fonction, on optimise l'efficacité des opérations de manutention, ce qui se traduit par un gain de temps et une réduction des coûts d'exploitation. Le mécanisme de pondération du temps de déplacement vertical permet de privilégier les placements qui favorisent l'accès rapide aux conteneurs urgents, assurant ainsi une meilleure fluidité des opérations et le respect des délais de livraison.

### 3.4.2 L'algorithme "Normal" (Recherche exhaustive)

L'algorithme "Normal" implémente une recherche exhaustive de toutes les combinaisons de placements de conteneurs possibles. Il explore systématiquement toutes les solutions valides, c'est-à-dire les placements qui respectent les contraintes du problème (nombre de conteneurs de chaque priorité, capacité des rangées, etc.). Pour chaque solution, il calcule le score *fitness* à l'aide de la fonction *Fitness* et conserve la meilleure solution trouvée. La Figure 3.12 illustre le fonctionnement de cet algorithme.

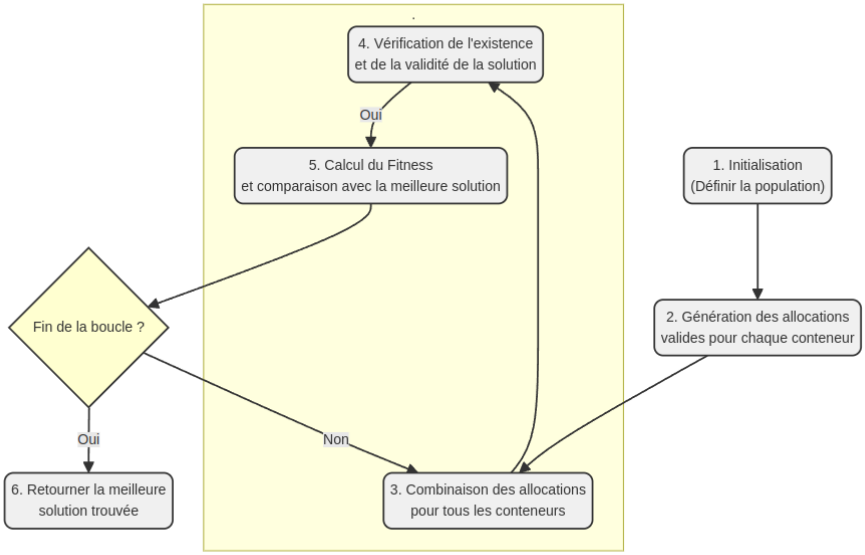



FIGURE 3.12 – Schéma du fonctionnement de l'algorithme "Normal" (Recherche exhaustive)

Le processus se déroule comme suit :

1. **Génération des allocations valides :** L'algorithme commence par identifier toutes les allocations possibles pour chaque conteneur, en tenant compte de sa priorité et des emplacements disponibles dans les rangées. Cette étape est réalisée par la fonction *SetRankAllocations*.
2. **Combinaison des allocations :** Ensuite, l'algorithme génère toutes les combinaisons possibles d'allocations pour l'ensemble des conteneurs entrants. Chaque combinaison représente une solution de placement potentielle.
3. **Évaluation des solutions :** Pour chaque solution générée, l'algorithme calcule le score *fitness* à l'aide de la fonction *Fitness*.
4. **Sélection de la meilleure solution :** L'algorithme conserve la solution ayant le score *fitness* le plus bas, c'est-à-dire le temps de manutention total le plus court.

### 3.4.3 Biogeography-Based Optimization (BBO)

— **Application de BBO au contexte du placement des conteneurs :**
La figure 3.13 illustre le fonctionnement de l'algorithme BBO appliqué au problème de placement des conteneurs.

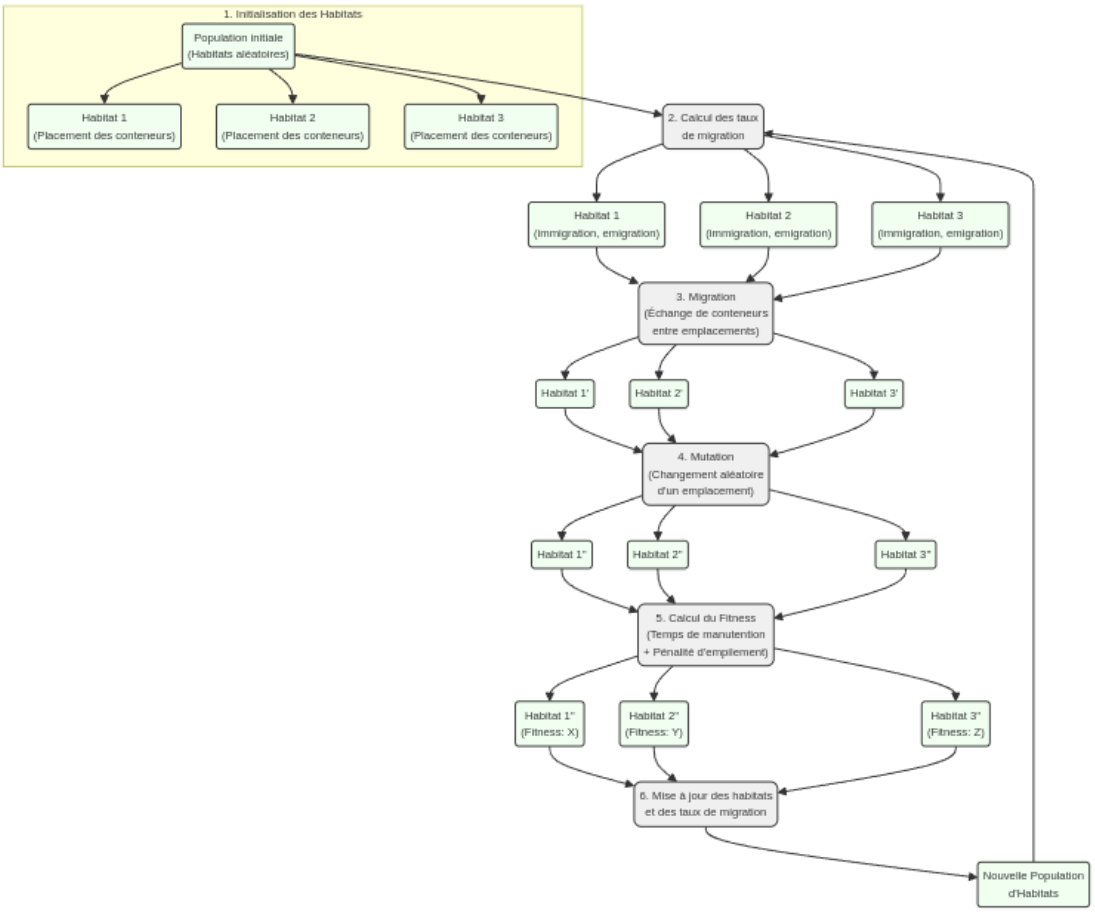
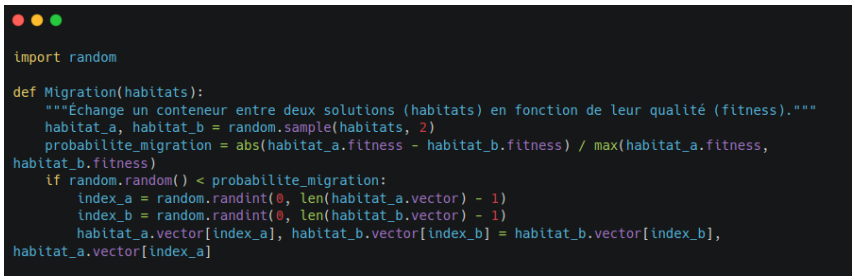



FIGURE 3.13 – Schéma du fonctionnement de l'algorithme BBO pour le placement des conteneurs

**Habitats (Solutions possibles) :** Chaque habitat représente une solution potentielle au problème du placement des conteneurs. Un habitat est défini comme une configuration spécifique de placement des conteneurs dans le bloc de stockage.
**Îles (Conteneurs) :** Chaque conteneur entrant est considéré comme une "île" qui doit être placée dans un habitat (solution).
**SIV (Variables de placement) :** Les SIV (Suitability Index Variables) représentent les caractéristiques d'un habitat qui influencent la "suitabilité" de l'habitat pour un conteneur donné. Dans notre cas, les SIV correspondent à la position du conteneur dans le bloc de stockage.

L'algorithme BBO utilise deux mécanismes principaux pour explorer l'espace des solutions :

**Migration :** La migration simule le mouvement des espèces entre les habitats. Dans notre contexte, cela correspond à l'échange d'informations entre les solutions de placement. Des conteneurs (îles) peuvent être déplacés d'une solution (habitat) vers une autre, en fonction de la "suitabilité" des habitats. La probabilité de migration d'un conteneur est influencée par la priorité du conteneur et par la différence de "suitabilité" entre les habitats. La figure 3.14 illustre ce processus avec un code simplifié.

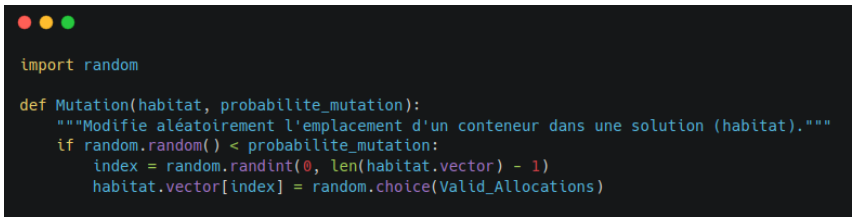


```python
import random

def Migration(habitats):
    """Échange un conteneur entre deux solutions (habitats) en fonction de leur qualité (fitness)."""
    habitat_a, habitat_b = random.sample(habitats, 2)
    probabilite_migration = abs(habitat_a.fitness - habitat_b.fitness) / max(habitat_a.fitness,
habitat_b.fitness)
    if random.random() < probabilite_migration:
        index_a = random.randint(0, len(habitat_a.vector) - 1)
        index_b = random.randint(0, len(habitat_b.vector) - 1)
        habitat_a.vector[index_a], habitat_b.vector[index_b] = habitat_b.vector[index_b],
habitat_a.vector[index_a]
```

FIGURE 3.14 – Illustration simplifiée de la fonction migration dans BBO

**Mutation :** La mutation représente des changements aléatoires dans les caractéristiques d'un habitat. Dans notre cas, cela correspond à des modifications aléatoires de la position des conteneurs dans une solution. La mutation permet d'introduire de la diversité dans la population de solutions et d'explorer de nouvelles régions de l'espace des solutions. La figure 3.15 montre un exemple simplifié de mutation.



```python
import random

def Mutation(habitat, probabilite_mutation):
    """Modifie aléatoirement l'emplacement d'un conteneur dans une solution (habitat)."""
    if random.random() < probabilite_mutation:
        index = random.randint(0, len(habitat.vector) - 1)
        habitat.vector[index] = random.choice(Valid_Allocations)
```

FIGURE 3.15 – Illustration simplifiée de la fonction mutation dans BBO

L'adaptation de BBO à notre problème de placement de conteneurs est illustrée dans le code source par les fonctions *Migrate* et *Mutation*. La fonction *Migrate* gère l'échange de conteneurs entre les solutions en utilisant les taux d'immigration et d'émigration qui sont calculés en fonction du score *fitness* de chaque solution. La fonction *Mutation* modifie aléatoirement la position d'un conteneur dans une solution avec une probabilité déterminée par le paramètre *mutation*.

### 3.4.4 L'algorithme Génétique (AG)

— **Caractéristique de l'algorithme Génétique :**

* **Chromosome** : Représentation des solutions potentielles sous forme de chaînes de gènes. Chaque gène représente une caractéristique ou une variable de la solution.
* **Population initiale** : Ensemble aléatoire de chromosomes générés au début de l'algorithme. La taille de la population influence la diversité et la qualité des solutions.
* **Sélection** : Processus pour choisir les chromosomes les plus adaptés en fonction de leur fitness. Les méthodes courantes incluent la sélection par roulette, par tournoi, ou par rang.
* **Croisement (Crossover)** : Combinaison de deux chromosomes parentaux pour créer de nouveaux chromosomes (descendants). Le croisement permet d'explorer de nouvelles solutions en combinant les bonnes caractéristiques des parents.

* **Mutation** : Modification aléatoire de certains gènes des chromosomes pour introduire de la diversité dans la population. La mutation aide à éviter la convergence prématurée vers une solution suboptimale.

— **Adaptation du AG au problème spécifique du placement des conteneurs en intégrant la prédiction de priorité :**
Dans notre application, le AG est adapté au problème du placement des conteneurs en tenant compte de la prédiction de priorité de chaque conteneur. La figure 3.16 illustre le fonctionnement de cet algorithme.

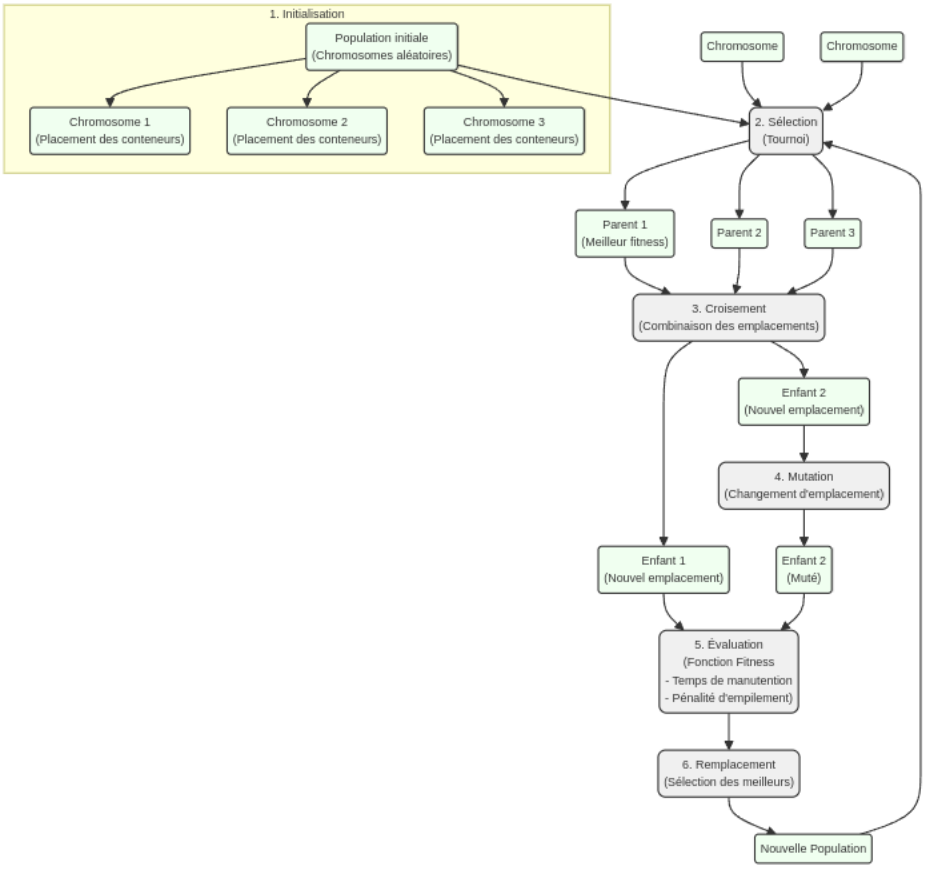


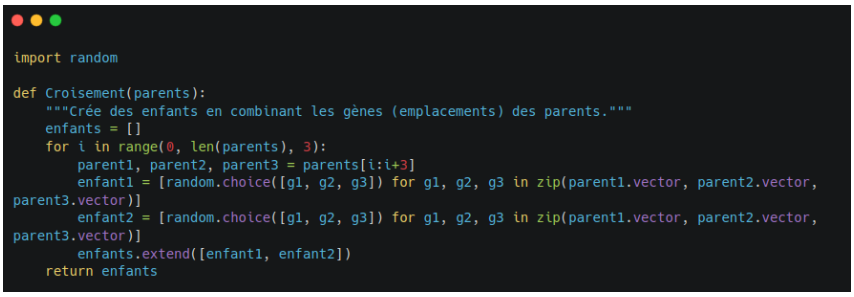

FIGURE 3.16 – Schéma du fonctionnement de l'algorithme génétique pour le placement des conteneurs

* **Représentation des solutions (chromosomes) et des gènes :** Un chromosome représente une solution de placement potentielle. Chaque gène du chromosome correspond à un conteneur et sa valeur représente la position du conteneur dans le bloc de stockage (rangée et niveau).
* **Choix des opérateurs de sélection, de croisement et de mutation :** Le code source utilise la sélection par tournoi pour choisir les parents (voir Figure 3.17). Le croisement est implémenté par la fonction *Croisement* qui combine les gènes de trois parents pour créer deux enfants (voir Figure 3.18). La mutation est implémentée par la fonction *Mutation* qui modifie aléatoirement la position d'un conteneur dans un chromosome (voir Figure 3.19).

```python
import random

def Selection(population, nombre_parents):
    """Sélectionne les parents pour la reproduction via des tournois."""
    parents = []
    for _ in range(nombre_parents):
        tournoi = random.sample(population, k=3)
        parents.append(min(tournoi, key=lambda chromosome: chromosome.fitness))
    return parents
```

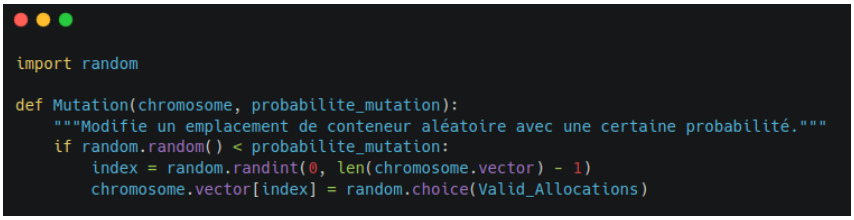
FIGURE 3.17 – Illustration simplifiée de la sélection par tournoi dans l'algorithme génétique



```python
import random

def Croisement(parents):
    """Crée des enfants en combinant les gènes (emplacements) des parents."""
    enfants = []
    for i in range(0, len(parents), 3):
        parent1, parent2, parent3 = parents[i:i+3]
        enfant1 = [random.choice([g1, g2, g3]) for g1, g2, g3 in zip(parent1.vector, parent2.vector,
parent3.vector)]
        enfant2 = [random.choice([g1, g2, g3]) for g1, g2, g3 in zip(parent1.vector, parent2.vector,
parent3.vector)]
        enfants.extend([enfant1, enfant2])
    return enfants
```

FIGURE 3.18 – Illustration simplifiée du croisement dans l'algorithme génétique



```python
import random

def Mutation(chromosome, probabilite_mutation):
    """Modifie un emplacement de conteneur aléatoire avec une certaine probabilité."""
    if random.random() < probabilite_mutation:
        index = random.randint(0, len(chromosome.vector) - 1)
        chromosome.vector[index] = random.choice(Valid_Allocations)
```

FIGURE 3.19 – Illustration simplifiée de la mutation dans l'algorithme génétique

## 3.5 Illustration des performances des algorithmes d'optimisation

Afin d'illustrer concrètement le fonctionnement et l'efficacité des algorithmes d'optimisation, nous les appliquons à un exemple simplifié de placement de conteneurs. Supposons qu'un navire arrive au port avec 17 conteneurs à décharger, répartis selon les priorités suivantes :
Priorité 1 : 3 conteneurs
Priorité 2 : 6 conteneurs
Priorité 3 : 3 conteneurs
Priorité 4 : 3 conteneurs
Priorité 5 : 2 conteneurs

La figure 3.20 présente la configuration initiale du terminal à conteneurs avant l'arrivée de ces nouveaux conteneurs.

Nous appliquons ensuite les trois algorithmes d'optimisation étudiés précédemment – "Normal" (recherche exhaustive), BBO (Biogeography-Based Optimization) et AG (Algorithme Génétique) – à ce problème de placement. Le nombre d'itérations pour chaque

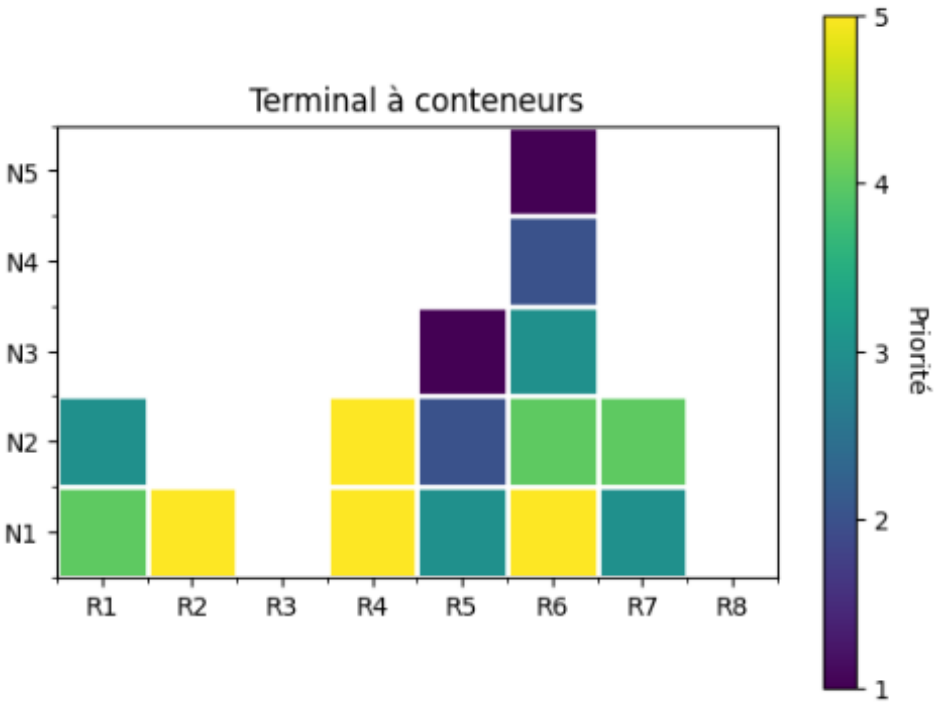



FIGURE 3.20 – État initial du terminal à conteneurs.

algorithme est limité à 150. Le tableau 3.5 résume les performances obtenues par chaque algorithme, en termes de *fitness* (représentant le temps de manutention total) et de temps de calcul.

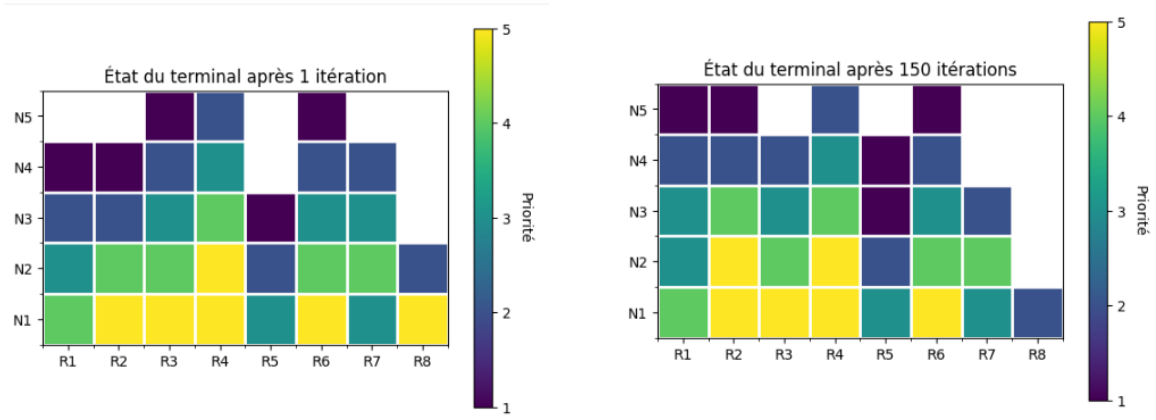
TABLE 3.5 – Comparaison des performances des algorithmes d'optimisation.

| **Algorithme** | **Fitness** | **Temps de calcul (s)** |
|---|---|---|
| Normal | 92.725 | 0.033 |
| BBO | 90.533 | 0.351 |
| AG | 86.647 | 1.117 |

La figure 3.21 illustre visuellement l'évolution de la solution trouvée par l'algorithme AG au cours des itérations. La figure (a) représente l'état du terminal après une seule itération du AG, tandis que la figure (b) montre l'état du terminal après 150 itérations. On observe une nette amélioration de la disposition des conteneurs, passant d'une configuration relativement désordonnée après une itération à une configuration optimisée après 150 itérations.

## 3.6 Conclusion : Une approche hybride pour un placement intelligent des conteneurs

Ce chapitre a présenté une solution hybride pour l'optimisation du placement des conteneurs dans les terminaux portuaires. Cette solution repose sur deux piliers : la prédiction de la priorité des conteneurs à l'aide de modèles d'apprentissage automatique et l'application d'algorithmes bio-inspirés pour optimiser leur placement.




(a) Après 1 itération.

(b) Après 150 itérations.

FIGURE 3.21 – Évolution de la solution obtenue par l'algorithme AG.

L'évaluation comparative de trois algorithmes d'apprentissage automatique (Random Forest, Gradient Boosting et XGBoost) a mis en évidence la supériorité de XGBoost en termes de précision de prédiction. L'intégration des prédictions de priorité dans les algorithmes de placement a ensuite été explorée, en utilisant trois approches : "Normal" (recherche exhaustive), Biogeography-Based Optimization (BBO) et Algorithme Génétique (AG).

L'analyse comparative, basée sur la fonction *Fitness* qui modélise le temps total de manutention, a révélé que l'algorithme génétique (AG) fournit les solutions les plus performantes. De plus, l'AG a démontré sa capacité à optimiser non seulement le temps de manutention, mais aussi l'utilisation de l'espace dans le terminal.

Cette approche hybride offre une vision globale et intégrée de l'optimisation du placement des conteneurs, en tenant compte de la priorité des conteneurs, des temps de déplacement de la grue et de l'utilisation de l'espace. Elle ouvre ainsi la voie à une gestion plus efficiente et performante des terminaux portuaires, permettant de répondre aux défis croissants du trafic conteneurisé mondial.

# Conclusion Générale

La problématique cruciale du placement des conteneurs dans les terminaux portuaires représente un défi majeur pour la fluidité des opérations, l'optimisation de l'espace et la satisfaction des clients. Face à la complexité croissante de cette tâche, nous avons jugé nécessaire de définir la priorité des conteneurs de manière plus précise, car celle-ci a toujours été affectée de manière aléatoire. C'est pourquoi, dans notre travail, nous avons proposé une solution hybride basée sur les techniques de machine learning pour la prédiction de cette priorité et les techniques d'optimisation bio-inspirées pour détecter le meilleur emplacement. Nous avons utilisé trois algorithmes de machine learning (RandomForestClassifier, GradientBoostingClassifier, XGBoostClassifier) et avons trouvé que le XGBoostClassifier donnait les meilleurs résultats. Cette priorité a ensuite été utilisée par les algorithmes bio-inspirés (algorithme génétique, BBO, recherche exhaustive) et nous avons trouvé que le meilleur algorithme était l'algorithme génétique.

L'approche hybride, adaptable à divers contextes et contraintes opérationnelles, ouvre des perspectives prometteuses pour une gestion plus performante et efficiente des terminaux à conteneurs. Ce potentiel d'amélioration peut être encore renforcé en explorant les pistes suivantes :

— **Amélioration du modèle de prédiction** : L'exploration de techniques d'apprentissage automatique plus avancées et l'intégration de données en temps réel (trafic portuaire, conditions météorologiques) permettraient d'améliorer la précision des prédictions de temps de séjour.

— **Intégration de contraintes supplémentaires** : La prise en compte de contraintes opérationnelles spécifiques, telles que les types de conteneurs, les restrictions d'empilage et les poids, enrichirait la fonction *Fitness* des algorithmes et permettrait de générer des solutions plus réalistes et plus performantes.

— **Optimisation multi-objectifs** : L'extension de l'algorithme pour optimiser simultanément plusieurs objectifs (temps de manutention, utilisation de l'espace, relocations) permettrait de répondre aux besoins complexes des terminaux modernes.

— **Hybridation des algorithmes** : La combinaison des forces de différents algorithmes d'optimisation, comme l'utilisation de BBO pour la diversification des solutions et du AG pour l'affinement, pourrait conduire à des solutions encore plus performantes.

L'application de cette solution dans les ports, en tenant compte de ses spécificités, nécessiterait :

— **Réaliser une analyse approfondie des infrastructures et des systèmes d'information du port** : Cette analyse permettra d'évaluer précisément les prérequis

techniques, les besoins en ressources humaines et les coûts d'implémentation de la solution.

— **Développer une stratégie de mise en œuvre progressive** : Une approche par étapes, en commençant par un projet pilote sur une zone restreinte du yard, permettrait de tester la solution, d'identifier les difficultés potentielles et d'ajuster les paramètres avant un déploiement à plus grande échelle.

— **Investir dans la formation du personnel** : La réussite de la mise en œuvre dépendra de l'adhésion et de la compétence du personnel. Des formations dédiées à l'utilisation de la solution et aux concepts d'apprentissage automatique seront essentielles.

— **Explorer le potentiel des nouvelles technologies** : L'intégration de l'Internet des Objets (IoT) et du Big Data pourrait améliorer la performance de la solution en fournissant des données en temps réel sur les mouvements des conteneurs et les conditions du terminal.

# Bibliographie


[1] Mohammadi A. Akbari, R. and K. Ziarati. A novel bee swarm optimization algorithm for numerical function optimization. *Communications in Nonlinear Science and Numerical Simulation*, 15(10) :3142–3155, 2010.

[2] Maria Ayala. *Programmation linéaire en nombres entiers pour l'ordonnancement cyclique sous contraintes de ressources*. Theses, Université Paul Sabatier - Toulouse III, June 2011.

[3] L Breiman. Random forests. *Machine learning*, 45(1) :5–32, 2001.

[4] C Pal C Beckham. A simple squared-error reformulation for ordinal classification. *arXiv preprint arXiv :1612.00775*, 2016.

[5] Nibedita Chakrabarty, Tanaya Kundu, Sucharita Dandapat, and Anirban Sarkar. Flight arrival delay prediction using gradient boosting classifier. In *Emerging Technologies in Data Mining and Information Security*, pages 573–580. Springer, 2019.

[6] R. Chelouah and P. Siarry. Tabu search applied to global optimization. *European Journal of Operational Research*, 123(2) :256–270, 2000.

[7] K Das, J Jiang, and JNK Rao. Mean squared error of empirical predictor. *The Annals of Statistics*, 32(2) :818–840, 2004.

[8] Birattari M. Dorigo, M. and T. Stutzle. Ant colony optimization. *IEEE Computational Intelligence Magazine*, 1(4) :28–39, 2006.

[9] Mohamed Elwakil, Mohamed S. Gheith, and Amr B. Eltawil. A new hybrid salp swarm-simulated annealing algorithm for the container stacking problem. In *International Conference on Operations Research and Enterprise Systems*, 2020.

[10] Marko Gulić, Livia Maglić, Tomislav Krljan, and Lovro Maglić. Solving the container relocation problem by using a metaheuristic genetic algorithm. *Applied Sciences*, 12(15), 2022.

[11] TO Hodson, TM Over, and SS Foks. Mean squared error, deconstructed. *Journal of Advances in Modeling Earth Systems*, 2021.

[12] Jiahuan Jin, Mingyu Ma, Huan Jin, Tianxiang Cui, and Ruibin Bai. Container terminal daily gate in and gate out forecasting using machine learning methods. *Transport Policy*, 132 :163–174, 2023.

[13] Nobar Kassabian. *Optimisation du stockage des conteneurs dans un terminal portuaire*. Theses, Université de Haute Alsace - Mulhouse, December 2022.

[14] Gelatt C.D. Kirkpatrick, S. and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598) :671–680, 1983.

[15] Ioanna Kourounioti, Amalia Polydoropoulou, and Christos Tsiklidis. Development of models predicting dwell time of import containers in port container terminals – an

artificial neural networks application. *Transportation Research Procedia*, 14 :243–252, 2016. Transport Research Arena TRA2016.

[16] MP LaValley. Logistic regression. *Circulation*, 117(18) :2395–2399, 2008.

[17] David R. Morrison, Sheldon H. Jacobson, Jason J. Sauppe, and Edward C. Sewell. Branch-and-bound algorithms : A survey of recent advances in searching, branching, and pruning. *Discrete Optimization*, 19 :79–102, 2016.

[18] Muzammal Nawaz. Container tracking data.xlsx, 2024.

[19] Mahesh Pal. Random forest classifier for remote sensing classification. *International journal of remote sensing*, 26(1) :217–222, 2005.

[20] LE Peterson. K-nearest neighbor. *Scholarpedia*, 4(2) :1883, 2009.

[21] NGN Prasad and JNK Rao. The estimation of the mean squared error of small-area estimators. *Journal of the American Statistical Association*, 85(409) :163–171, 1990.

[22] R Beresford S Agatonovic-Kustrin. Basic concepts of artificial neural network (ann) modeling and its application in pharmaceutical research. *Journal of pharmaceutical and biomedical analysis*, 22(5) :717–727, 2000.

[23] Jean-François Scheid. *Programmation linéaire. Méthodes et applications*. 10 2015.

[24] Shishir Kumar Shandilya, Agni Datta, and Atulya K. Nagar. *Nature-inspired Algorithms*, pages 3–36. Springer Nature Singapore, Singapore, 2023.

[25] D. Simon. Biogeography-based optimization. *IEEE Transactions on Evolutionary Computation*, 12(6) :702–713, 2008.

[26] S.N. Sivanandam and S.N. Deepa. *Introduction to genetic algorithms*. Springer, 2008.

[27] C Guestrin T Chen. Xgboost : A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM, 2016.

[28] Ruoqi Wang, Jiawei Li, and Ruibin Bai. Prediction and analysis of container terminal logistics arrival time based on simulation interactive modeling : A case study of ningbo port. *Mathematics*, 11(15), 2023.

[29] Tan D. Wang, D. and L. Liu. Particle swarm optimization algorithm : an overview. *Soft computing*, 22(2) :387–408, 2018.

[30] S Weisberg. *Applied linear regression*. John Wiley & Sons, 2005.

[31] Cheng-Hong Yang and Po-Yin Chang. Forecasting the demand for container throughput using a mixed-precision neural architecture based on cnn–lstm. *Mathematics*, 8(10), 2020.

[32] Canrong Zhang, Hao Guan, Yifei Yuan, Weiwei Chen, and Tao Wu. Machine learning-driven algorithms for the container relocation problem. *Transportation Research Part B : Methodological*, 139 :102–131, 2020.

**Résumé**

Ce mémoire propose une approche hybride intelligente base sur les technique de l'intelligence artificielle pour améliorer le stockage des conteneurs dans les terminaux portuaires, améliorant ainsi la fluidité des opérations, la réduction des temps de manutention et l'utilisation efficace de l'espace. Face aux limitations des méthodes traditionnelles, cette approche combine un modèle de prédiction de la priorité des conteneurs, basé sur l'apprentissage automatique, avec des algorithmes bio-inspirés pour détecter le meilleur emplacement. L'intégration de la prédiction de priorité permet de prioriser les conteneurs à sortie rapide et d'optimiser leur placement dans le yard de stockage. Les simulations effectuées démontrent l'efficacité de cette approche, notamment grâce à l'algorithme génétique qui a surpassé les autres méthodes testées. Cette solution hybride, adaptable à divers contextes portuaires, ouvre des perspectives prometteuses pour une gestion plus performante et efficiente des terminaux à conteneurs à l'échelle mondiale.

**Mots clés :** Transport maritime, Priorité du conteneur, LSTM, Optimisation, Algorithmes Bio-inspirés.

**Abstract**

This study proposes a hybrid intelligent approach based on artificial intelligence techniques to improve container storage in port terminals, thereby improving operational fluidity, reducing handling times, and enabling efficient space utilization. Addressing the limitations of traditional methods, this approach combines a machine learning-based container priority prediction model with bio-inspired algorithms to identify optimal storage locations. Integrating priority prediction allows for prioritizing containers with quick turnaround times and optimizing their placement within the storage yard. Simulations demonstrate the effectiveness of this approach, particularly with the genetic algorithm outperforming other tested methods. This hybrid solution, adaptable to various port contexts, offers promising prospects for more efficient and effective management of container terminals globally.

**Keywords :** Maritime Transport, Container Priority, LSTM, Optimization, Bio-inspired Algorithms.