

# Rapport — Complement\_IA : Bataille Navale et heuristiques

HAMLIL, Mohamed Ali Tewfik  
SGHIOUAR IDRISSE, Saad

24 novembre 2025

## Résumé

Ce rapport présente le jeu étudié, les règles, les stratégies heuristiques implémentées, la démarche expérimentale suivie pour obtenir les résultats, une analyse des résultats et une conclusion.

## Table des matières

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Définition du jeu et règles</b>	<b>2</b>
<b>3</b>	<b>Revue de littérature</b>	<b>2</b>
<b>4</b>	<b>Stratégies étudiées (heuristiques)</b>	<b>3</b>
4.1	Uniforme (Uniform)	3
4.2	Markov	3
4.3	Monte Carlo	3
4.4	Smart / Hybride	3
<b>5</b>	<b>Analyse des résultats</b>	<b>4</b>
5.1	Tournoi	4
5.1.1	Procédure	4
5.1.2	Tableau du tournoi	4
5.1.3	Analyse du tournoi	4
5.2	Self-play	5
5.2.1	Procédure	5
5.2.2	Distribution (gaussienne) et visualisation	5
5.2.3	Tableau des performances (moyennes de coups)	5
5.2.4	Analyse des moyennes de coups	5
<b>6</b>	<b>Conclusion</b>	<b>6</b>

# 1 Introduction

Ce document présente une étude expérimentale portant sur l’implémentation du jeu de la *Bataille Navale*. L’objectif principal est d’analyser et de comparer plusieurs stratégies de tir (principalement heuristiques) afin d’évaluer leurs performances selon différents indicateurs : taux de victoire, efficacité moyenne (nombre de coups nécessaires), stabilité des résultats et comportement selon les configurations de jeu.

L’ensemble du projet est organisé comme suit :

- **Code source** : disponible dans le répertoire `src/`, comprenant les modules de simulation, les stratégies testées et les outils d’expérimentation.
- **Résultats expérimentaux** : les fichiers générés (CSV) sont automatiquement déposés dans `Results/`. Ils regroupent les données exploitées pour l’analyse : résumés de tournois, mesures de self-play, statistiques de performance, etc.
- **Documentation technique** : accessible dans `docs/`, elle inclut la description des classes, modules et structures internes du projet. Selon le format final du rapport (PDF ou HTML), ces pages peuvent être référencées ou intégrées sous forme de liens.

L’étude qui suit repose sur une série d’expérimentations reproductibles, réalisées à partir des données contenues dans ce dépôt. Elle vise à mettre en évidence les différences de comportement entre stratégies, à dégager des tendances générales et à fournir une base comparative solide pour de futurs développements.

## 2 Définition du jeu et règles

La *Bataille Navale* est un jeu à deux joueurs se déroulant sur une grille carrée. Chaque joueur dispose d’une flotte composée de navires de longueurs prédéfinies et cherche à détruire la flotte adverse à l’aide de tirs successifs. Les hypothèses utilisées dans ce projet sont les suivantes :

- La grille est de taille  $N \times N$  (souvent  $N = 10$ ).
- La flotte est composée de navires de longueurs fixes, par exemple  $\{5, 4, 3, 3, 2, 2\}$ .
- Les navires sont placés intégralement dans la grille, sans chevauchement, et orientés horizontalement ou verticalement.
- Les joueurs tirent à tour de rôle sur des cases encore non visées. Chaque tir est classé comme *touché*, *coulé* ou *à l’eau*.
- La partie se termine lorsqu’un joueur a coulé tous les navires adverses.

Les variantes éventuelles (taille de grille différente, flotte réduite, règles alternatives) sont précisées dans la section méthodologique du rapport.

## 3 Revue de littérature

Cette section présente une synthèse des principales approches étudiées dans la littérature concernant la résolution du jeu de la *Bataille Navale*. La version complète, incluant une bibliographie détaillée, est disponible dans : `rapport/revue_litterature`.

### Résumé

Les travaux existants peuvent être regroupés en trois grandes familles :

- **Approches heuristiques** : stratégies de recherche de type *hunt-target*, utilisation de motifs de parité pour optimiser le recouvrement de la grille, balayages systématiques améliorés, ainsi que diverses règles dérivées de la géométrie des navires.
- **Méthodes probabilistes** : élaboration de matrices de probabilité associant à chaque case la vraisemblance de présence d’un navire, actualisation des estimations après chaque

tir, ou encore stratégies reposant sur des distributions construites à partir d'échantillonnages ou de simulations répétées.

- **Modèles computationnels avancés** : techniques d'exploration fondées sur des arbres de recherche, stratégies hybrides combinant heuristiques et estimations probabilistes, méthodes stochastiques basées sur la génération de nombreux scénarios de placement, ainsi que diverses formes d'optimisation algorithmique.

La littérature souligne que les stratégies combinant heuristiques et informations probabilistes sont particulièrement efficaces pour la phase de *chasse*, alors que les méthodes reposant sur des mécanismes d'exploration et d'évaluation systématique améliorent notablement la phase de *ciblage*. Les performances observées dépendent cependant de plusieurs paramètres : taille de la grille, composition de la flotte, contraintes de placement et quantité de simulations utilisées.

La bibliographie complète est fournie dans le dossier : `rapport/revue_litterature/`.

## 4 Stratégies étudiées (heuristiques)

Les stratégies implémentées dans le dépôt reposent sur deux composantes principales :

- **Partie déterministe (Target)** : dédiée au *ciblage* — exploitation des informations directes (cases touchées) pour déterminer l'orientation du navire et le couler.
- **Partie probabiliste (Hunt)** : destinée à l'exploration et à l'estimation des positions plausibles des navires (placement counting, Monte Carlo) pour orienter les tirs en phase de recherche.

Les principales heuristiques étudiées sont résumées ci-dessous.<sup>1</sup>

### 4.1 Uniforme (Uniform)

Stratégie de base entièrement aléatoire. À chaque tour, elle choisit uniformément une case non encore visée. Elle sert de référence minimale (baseline) pour comparer les approches plus élaborées.

### 4.2 Markov

Stratégie probabiliste fondée sur le *placement counting* : pour chaque case, l'algorithme compte le nombre de placements possibles de la flotte compatibles avec l'état du plateau (touchés, coulés, cases à l'eau). La case ayant la probabilité estimée la plus élevée d'appartenir à un navire est sélectionnée.

### 4.3 Monte Carlo

Méthode d'échantillonnage : un grand nombre de placements aléatoires de la flotte encore possible sont générés, en respectant toutes les contraintes observées. Une *heatmap* est ensuite construite en comptabilisant la fréquence d'occupation de chaque case dans ces échantillons. La stratégie tire sur la case la plus fréquemment occupée.

### 4.4 Smart / Hybride

Stratégie hybride combinant les deux composantes décrites ci-dessus : la phase d'exploration *Hunt* (probabiliste) et la phase de ciblage *Target* (déterministe). Concrètement :

- **Hunt (exploration probabiliste)** : la stratégie utilise des estimations (par placement counting ou Monte Carlo) pour repérer des zones prometteuses.
- **Target (ciblage déterministe)** : dès qu'une case est touchée, la stratégie applique un ciblage déterministe des cases adjacentes pour identifier l'orientation et couler le navire.

---

1. Les différentes heuristiques se trouvent dans : `src/heuristic/` et `src/joueurs/SmartBot.java`.

## 5 Analyse des résultats

Cette section présente et interprète les résultats issus des différentes campagnes d'expérimentation.

Deux types d'analyses ont été réalisés :

- *Tournoi* : permet une comparaison croisée des stratégies.<sup>2</sup>
- *Self-play* : vise à mesurer la stabilité et la distribution des performances d'une stratégie lorsqu'elle joue contre elle-même.<sup>3</sup>

### 5.1 Tournoi

#### 5.1.1 Procédure

Le tournoi consiste à faire s'affronter les différentes stratégies sur un grand nombre de parties selon la configuration expérimentale définie dans la méthodologie.<sup>4</sup> Les résultats sont ensuite regroupés dans deux fichiers principaux :

- `tournament_summary.csv` : récapitulatif global (nombre de parties, victoires, taux de victoire, erreur standard) ;
- `tournament_pairwise.csv` : résultats détaillés des confrontations deux-à-deux.

#### 5.1.2 Tableau du tournoi

Le tableau suivant est trié par taux de victoire décroissant.<sup>5</sup>

Bot	Parties	Victoires	TauxVictoire	ErreurStd	Classement
Markov	300	223	0.743333	0.025218	1
MonteCarlo	300	201	0.670000	0.027148	2
Intelligent	300	170	0.566667	0.028610	3
Uniforme	300	6	0.020000	0.008083	4

#### 5.1.3 Analyse du tournoi

La meilleure stratégie observée dans ce tournoi est **Markov**, avec un taux de victoire de 0.743333. Il convient d'interpréter ce résultat en conjonction avec l'erreur standard afin d'évaluer la robustesse statistique de cette performance.

Sur le plan du coût de calcul, les approches fondées sur des simulations Monte-Carlo s'avèrent généralement plus gourmandes en ressources, surtout lorsque le budget d'évaluation par coup est élevé. Ce coût doit être mis en balance avec le gain effectif en performance lors du choix d'une stratégie pour une application donnée.

Par ailleurs, les stratégies de type markovien tendent à reproduire une séquence initiale de coups identique tant qu'aucun navire n'est touché. Ce comportement déterministe sur les premiers coups peut induire des biais exploitables et mérite d'être analysé plus finement.

Les méthodes basées Monte-Carlo ne montrent pas nécessairement de motif décisionnel clair et leur comportement peut varier fortement selon la distribution des positions initiales des bateaux ; elles dépendent souvent de l'échantillonnage.

2. Un tournoi désigne ici un ensemble de confrontations répétées entre toutes les stratégies considérées.

3. Le self-play est couramment utilisé pour isoler la dynamique intrinsèque d'une stratégie, indépendamment d'un adversaire donné.

4. Voir la section Méthodologie pour la description des paramètres : nombre de parties, conditions de départ, structure du tir.

5. Ce tri permet d'identifier rapidement les stratégies dominantes dans le tournoi.

## 5.2 Self-play

### 5.2.1 Procédure

Les analyses de self-play visent à quantifier la performance intrinsèque d'une stratégie en l'opposant à elle-même<sup>6</sup>. Les résultats agrégés sont enregistrés dans `performance_summary.csv`.

### 5.2.2 Distribution (gaussienne) et visualisation

Lorsque la distribution des longueurs de parties s'approche d'une loi normale, une densité gaussienne peut être superposée pour lisibilité analytique.<sup>7</sup>

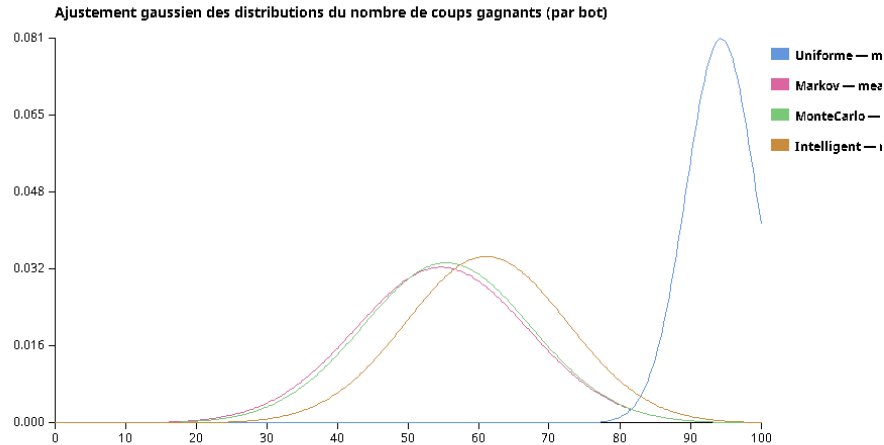


FIGURE 1 – Distribution (approx. gaussienne) des longueurs de parties pour un exemple de stratégie.

### 5.2.3 Tableau des performances (moyennes de coups)

Bot	Essais	MoyenneCoups	ErreurStd
Markov	100	54.770000	1.218449
MonteCarlo	100	55.480000	1.185304
Intelligent	100	61.130000	1.140056
Uniforme	100	94.340000	0.493845

### 5.2.4 Analyse des moyennes de coups

La stratégie la plus efficace, au sens du plus faible nombre moyen de coups nécessaires pour remporter une partie, est **Markov**, avec une moyenne de 54.770000<sup>8</sup>.

Les résultats issus du self-play confirment la cohérence interne de la stratégie : jouer contre soi-même met en évidence son comportement intrinsèque, indépendant de la nature de l'adversaire. Ces observations sont en accord avec les performances mesurées lors du tournoi, renforçant la fiabilité de l'évaluation globale.

6. Cette approche met en évidence la régularité d'une stratégie sans dépendre d'un adversaire particulier.

7. L'ajustement gaussien n'est informatif que si la distribution empirique présente un comportement unimodal et symétrique.

8. Comme pour les taux de victoire, l'erreur standard permet d'évaluer la stabilité de la performance.

## 6 Conclusion

Cette étude a permis d'évaluer de manière systématique plusieurs stratégies appliquées au jeu de la Bataille Navale, en combinant analyses de tournoi et expérimentations en self-play. Les résultats se déclinent comme suit :

### Observations principales

- Les approches fondées sur le *placement counting* (stratégie Markov) offrent les meilleures performances globales : taux de victoire élevé, efficacité moyenne supérieure et excellente stabilité statistique.
- Les méthodes Monte Carlo présentent des performances proches mais requièrent un coût de calcul plus élevé et montrent une plus grande variabilité liée à l'échantillonnage.
- La stratégie hybride combine efficacement ciblage déterministe et exploration probabiliste, se montrant robuste et cohérente avec les tendances observées.

### Pistes d'amélioration

- Optimiser le temps de calcul des approches Monte Carlo (algorithmes plus rapides ou parallélisation).
- Affiner la composante probabiliste via des modèles plus précis ou des heuristiques adaptées.
- Intégrer des stratégies apprenantes pour adapter la recherche à des configurations ou distributions de flotte variées.
- Étendre les expérimentations à d'autres tailles de grille ou schémas de placement afin d'obtenir une évaluation plus complète des comportements stratégiques.

## Références

- [1] Exemple Auteur. Monte carlo methods for battleship-like inference. *Journal d'exemple*, 2013.
- [2] Marie Dupont and Paul Martin. Probabilistic placement counting for search games. *Revue des Algorithmes Probabilistes*, 2010. Article de synthèse, disponible en préprint.
- [3] Donald E. Knuth. *The T<sub>E</sub>Xbook*. Addison-Wesley, 1984.
- [4] Stuart Russell and Peter Norvig. *Artificial Intelligence : A Modern Approach*. Prentice Hall, 2009.