AI Pêche aux Moules

Manuel Administrateur

KARAMI Anir • BUT3APP TPC

Objectif

Installer, configurer et exploiter le serveur + le client IA. Ce document sert de guide pratique pour lancer des parties, diagnostiquer les erreurs et maintenir un niveau de performance stable.

Dernière mise à jour : 25 janvier 2026

Table des matières

1 Vue d'ensemble

Résumé

Le projet comporte un **serveur Java** (gestion du labyrinthe, scoring, règles) et un **client IA** (prise de décision et envoi d'actions via TCP).

1.1 Architecture

```
Serveur (Java)  <--- socket TCP --->  Client IA (superAI)
```

2 Prérequis et installation

- Java JDK 8+ (recommandé : 11 ou 17)
- MG2D (lib graphique du serveur)
- Bash + Python 3 pour les scripts de tests

Conseil

Placez le dossier `MG2D/` au même niveau que `Serveur/` et compilez le serveur depuis `Serveur/` pour éviter les problèmes de classpath.

2.1 Installation MG2D

Le serveur dépend de MG2D pour l'affichage.

- Copier le dossier `MG2D/` https ://github.com/synave/MG2D à côté de `Serveur/`.
- Compiler depuis `Serveur/` (il détecte MG2D dans le dossier parent).

3 Structure du projet

```
IA/superAI/      Client IA
Serveur/         Serveur Java
MG2D/            Librairie graphique
tools/           Scripts de tests + seeds
mkdocs.yml       Documentation MkDocs
```

3.1 Fichiers clés (IA)

- `IA.java` : pipeline principal et orchestration.
- `IAActions.java` : génération/scoring des actions.
- `IACibles.java` : choix de cibles et adversaires.
- `IABeam.java` : beam search optionnel.

— `Distances.java` : BFS classique et bonus-aware.

4 Compilation

4.1 Serveur

```
cd Serveur
javac PecheAuxMoulesBoucle.java
```

4.2 Client IA

```
javac -d IA IA/superAI/*.java
```

5 Démarrage du serveur

Commande standard

```
java PecheAuxMoulesBoucle -nbJoueur 1 -delay 0 -timeout 3000
```

Port par défaut

Le serveur écoute par défaut sur le port 1337.

5.1 Paramètres serveur

Option	Rôle
`-nbJoueur`	Nombre de joueurs attendus
`-delay`	Délai entre tours (ms)
`-timeout`	Durée maximale d'une partie (ms)
`-numLaby`	Seed du labyrinthe
`-numPlacementBonus`	Seed des bonus
`-tauxDeMur`	Pourcentage de murs (0 à 50)

6 Seeds et reproductibilité

Principe

Fixer des seeds permet de comparer les performances entre versions de l'IA.

```
java PecheAuxMoulesBoucle -nbJoueur 1 -delay 0 -timeout 3000 \
  -numLaby 49811 -numPlacementBonus 52567
```

7 Lancement du client IA

```
java -cp IA superAI.ClientSuperAI 127.0.0.1 1337 MonEquipe
```

7.1 Aide rapide

```
java -cp IA superAI.ClientSuperAI -h
```

8 Protocole de communication (résumé)

Le serveur envoie un message texte par tour. Format :

```
LxH/structure/infosJoueurs
```

avec :

- `LxH` : taille du plateau,
- `structure` : `L*H` tokens séparés par `-`,
- `infosJoueurs` : `N-x,y- ...` (positions).

Le client répond par une action :

```
N | S | E | O | C | Bs-D | Bp-D1-D2-D3
```

Détail

Le serveur ajoute lui-même le préfixe joueur (`J0-`, `J1-`, etc.). Le client n'envoie que l'action.

8.1 Détail des tokens

- `Mu` = mur, `So` = sol,
- `Bs` = bonus saut, `Bp` = bonus trois pas,
- un nombre = moule (valeur en points, multiples de 10).

Le labyrinthe est toujours entouré de murs. La structure est fournie en parcours ligne par ligne (gauche à droite, haut en bas) et contient aussi la bordure de murs.

8.2 Positions des joueurs

Les coordonnées sont indexées à partir de 0. Par défaut :

- Joueur 0 : haut-gauche,
- Joueur 1 : bas-droite,
- Joueur 2 : haut-droite,
- Joueur 3 : bas-gauche.

8.3 Règles serveur (cas limites)

- Déplacement sur un mur : refusé (le joueur reste sur place).
- Bs : saut de 2 cases, bonus consommé uniquement si l'arrivée est non-mur.
- Bs sur mur : un pas simple est tenté, bonus conservé.
- Bs hors plateau : action ignorée.
- Bs ne ramasse que la case d'arrivée.
- Bp : trois pas successifs, directions invalides remplacées par `C`.
- Bp consomme le bonus si le joueur en possède un.
- Bp ramasse sur chaque case atteinte.
- Bs/Bp sans bonus : action ignorée.
- Direction Bs invalide : action ignorée.
- Plusieurs joueurs peuvent occuper la même case.
- Toute moule/bonus ramassé devient du sol (`So`).

9 Paramètres IA

Les paramètres sont passés en `cle=valeur`.

```
java -cp IA superAI.ClientSuperAI 127.0.0.1 1337 MonEquipe \
  modePlan=1 nbCiblesPlan=8 profondeurPlan=4
```

Preset par défaut

Le preset conseillé est **G1B0** (beam off, modules optionnels désactivés).

9.1 Familles de réglages

Catégorie	Exemples
Coûts	`coutSaut`, `coutTroisPas`, `penalitePasse`
Comportement	`penaliteRepet`, `penaliteAllerRetour`, `toursSansPointsMax`
Cibles	`verrouillageCible`, `seuilChangementCible`, `modeCompromis`
Planification	`modePlan`, `profondeurPlan`, `nbCiblesPlan`, `coeffProfondeur`
Beam search	`modeBeam`, `profondeurBeam`, `largeurBeam`, `seuilBeam`
Modules optionnels	`penaliteCibleAdverse`, `coeffCarteValeur`, `bonusUsageEfficace`

9.2 Paramètres sensibles (impact fort)

- `penaliteRepet` / `penaliteAllerRetour` : évitent les boucles.

— `modeBeam` + `profondeurBeam` : coût CPU vs stabilité.

— `modePlan` : stabilité des trajectoires, plus de calcul.

— `gainDistanceBonusMin` : usage des bonus (trop bas = gaspillage).

10 Presets disponibles

— **G1B0** : défaut, rapide et stable.

— **G1B0P** : même base + modules optionnels activés.

— **G1AUTO** : beam auto selon taille/ densité du labyrinthe.

11 Moteur IA (résumé technique)

Idées clés

— Distances via BFS (et BFS multi-états avec bonus).

— Score d'action = gains immédiats + estimation futur - pénalités.

— Mémoire anti-boucle (pénalités sur cycles courts et aller-retour).

— Planification locale (top-K) et beam search optionnel.

11.1 Pseudo-code simplifié

```
etat = parse(message)
dist = bfs(etat)
cible = choisir_cible(etat, dist)
pour chaque action:
  sim = simuler(action)
  score = evaluer(sim) - penalites + futur
choisir l'action au meilleur score
```

12 Outils de tests

```
AUTO_SERVER=1 ./tools/run_tests.sh 127.0.0.1 1337 MonEquipe tools/seeds.txt
```

Sorties

— `IA/superAI/logs/` : logs clients par preset

— `tools/results.csv` : agrégation des scores

Les fichiers de logs sont écrasés à chaque run (comportement voulu pour éviter les confusions).

12.1 Procédure de test standard

- Choisir un set de seeds stable (`tools/seeds.txt`).
- Lancer les presets (G1B0, G1B0P, G1AUTO).
- Comparer les tours moyens et la stabilité (éviter les boucles).
- Valider qu'aucun log ne signale d'action invalide.

12.2 Analyse des résultats

Le fichier `tools/results.csv` permet de trier par tours et points. Conserver les seeds et les résultats de référence pour détecter les régressions.

13 Diagnostic et supervision

- **Logs serveur** : erreurs de protocole, action invalide, fin de partie.
- **Logs client** : action choisie, position, bonus, top actions évaluées.
- **Boucles** : surveiller `va-et-vient` et `streak sans points`.

14 Maintenance et support

Checklist rapide

- Reproduire un bug avec une seed fixe.
- Sauvegarder le log serveur + log client correspondant.
- Vérifier la compilation (aucun `.class` obsolète).
- Rejouer les presets sur les seeds de référence.
- Mettre à jour la documentation si un paramètre change.

14.1 Archivage recommandé

- Garder une copie des logs serveurs importants.
- Conserver les meilleurs résultats (`results.csv`) par version.
- Noter les paramètres utilisés pour un run gagnant.

15 Documentation (MkDocs)

La documentation est générée avec MkDocs et publiée via GitHub Pages.

```
pip install -r requirements.txt
mkdocs serve
mkdocs gh-deploy --force --clean
```

16 Dépannage

Connexion refusée

Le serveur n'est pas démarré ou le port est incorrect.

ClassNotFound

Recompiler le client : `javac -d IA IA/superAI/*.java`.

Boucles longues

Baisser `penaliteRepet` / `penaliteAllerRetour` ou réduire la largeur du beam.

17 Glossaire

— **Seed** : numéro fixant le labyrinthe ou les bonus.

— **BFS** : parcours en largeur pour les distances.

— **Beam** : recherche par faisceau sur quelques coups futurs.

— **Cible** : case (moule/bonus) vers laquelle l'IA se dirige.

18 Ressources

— Repo : https://github.com/anirkm/AI_peche_aux_moules

— Doc : https://anirkm.github.io/AI_peche_aux_moules/