FireFighter vs Fire

Game

PROJET 01 : Multi-Agent Jeu Adversarial(Synchrone)

Groupe GL
- Hadil LADJ
- Mohamed Aziz BELHAJ HASSINE
- Malik MANSOUR

Plan de la Présentation

1. Introduction
2. Description du Jeu
3. Description de l'environnement (Grid)
4. Stratégie des Agents (Pompier)
5. Stratégie des Agents (Feu)
6. Statistiques et Résultats pour un environnement statique (Branche statique dans git)
7. l'environnement dynamique
8. Statistiques et Résultats pour un environnement Dynamique(Branche main dans git)
9. Organisation du Travail en groupe
10. Conclusion

01

INTRODUCTION

Qui contre Qui ? Ou ? Pourquoi ? Comment ?

INTRODUCTION

Fire vs FireFighter : est un jeu stratégique basé sur un système multi-agents, où deux types d'agents, le **"Fire Agent"** et le **"Fire Fighter Agent"**, s'affrontent dans un environnement. L'objectif principal pour chaque agent est d'atteindre un certain but pour augmenter le score, tout en interagissant avec un environnement et des éléments en temps réel en utilisant des stratégies spécifiques.

02

Description du Jeu

Description du Jeu

En général c'est un jeu adversarial **synchrone**, où les deux agents jouent tour par tour.

À chaque tour :

- Le **Fire Agent** se déplace pour brûler des objectifs en propageant du feu au même temps
- Le **Fire fighter Agent** intervient pour éteindre les incendies et protéger les objectifs.

Le but de chaque agent est d'avoir un score supérieur que celui de son adversaire.

- **Fire Agent (Feu) :** Gagne 1 point pour chaque forêt brûlée.
- **Fire Fighter Agent (Pompier) :** Gagne 1 point pour chaque forêt sécurisée.

Le jeu aura lieu dans un environnement spécial (Matrice ou Grid)

03

Description de l' Environnement

Répondre sur la question; Ou?

Structure de la Grid Statique

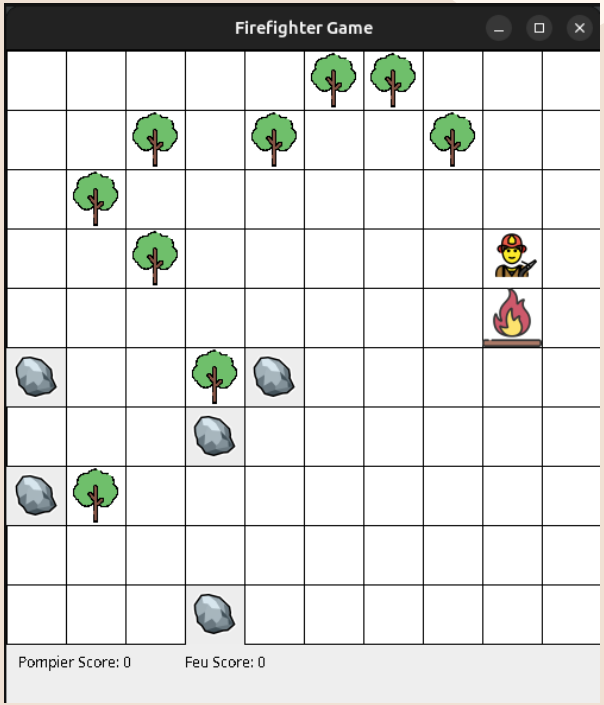
L'environnement du jeu est une matrice de 10*10, ou chaque case peut contenir un élément qui fait partie du jeu.

La classe **Grid** représente l'environnement du jeu, où chaque cellule peut contenir :

- **Forêts (Objectifs)** : Cases à protéger par le pompier ou à brûler par le feu.
- **Pierres (Barrières)** : Obstacles infranchissables pour les agents et la propagation du feu.
- **Eau (Cases sécurisées)** : Créées par le pompier pour stopper le feu.
- **Feu** : Zones enflammées qui se propagent aux cases voisines.
- **Cases vides** : Espaces libres pour les déplacements.

Environnement Statique

- La matrice est générée par la Classe **Grid** retourne un environnement statique qui est initié une seule fois et ne change pas ou sauf si les agents interagissent sur l'environnement.
- A chaque exécution la matrice doit contenir 9 Objectifs + 5 Barrières avec les 2 agents.
- Les éléments visuels du jeu sont représentés par des icônes : Fire **Agent => icône de flamme**, **Fire Fighter Agent => icône de pompier**.
- L'environnement est coloré en fonction de l'état des cases au début ils sont tous vide (case blanches) sauf les zones occupées, après les zones en brulé apparaissent en rouge et les zones protégées ou éteintes en bleu ciel.



04

Stratégie des Agents (Pompier)

Algorithme Greedy

Stratégie et Logique du Fire fighter Agent

1. Tache
   - Préserver toutes les cases et protéger les objectifs, à chaque objectifs protégé l'agent incrémente son score **"score++"**.
2. Stratégie de blocage :
   - Placer de l'eau autour des zones critiques pour empêcher la progression du feu avec la possibilité de l'extinction du feu.
   - Priorise les cases contenant des forêts pour les protéger en les sécurisant (eau).
3. Mouvement en diagonal :
   - Cet agent peut se bouger en diagonal car cet agent a 8 directions **UP, DOWN, LEFT, RIGHT UP_RIGHT, UP_LEFT, DOWN_RIGHT, DOWN_LEFT**
4. Algorithme de recherche :
   - Cet agent utilise l'algorithme greedy qui prend en compte non seulement l'objectif le plus proche à chaque étape, mais optimise également son chemin en considérant tous les objectifs à la fois

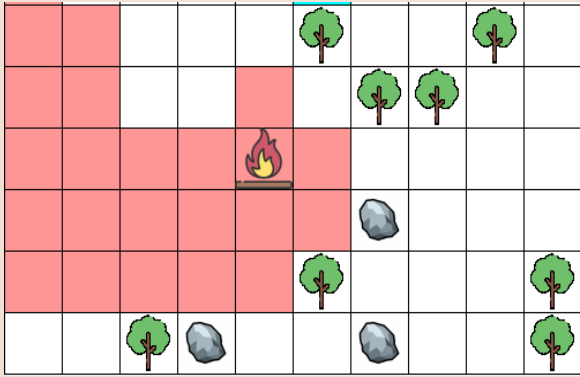
05

Stratégie des Agents (Feu)

Algorithme BFS

Stratégie et Logique du Fire Agent

- **Tache :**
  - Bruler le nombre maximal des forêts pour gagner le jeu, à chaque objectif bruler l'agent incrémente son score "score++"
- **Contournement du blocage et Propagation :**
  - Évite les cases sécurisées et les barrières.
  - **Déplacement aléatoire (jump)** Si le feu est bloqué, il saute vers une case libre choisie aléatoirement.
  - Cet agent a la possibilité de propager le feu dans les cases voisins
- **Mouvement normal :**
  - Cet agent peut bouger que dans 4 directions **UP, DOWN, LEFT, RIGHT.**
- **Algorithme de recherche :**
  - Cet agent utilise l'algorithme de recherche en largeur BFS, l'algo démarre de la position actuelle de l'agent toutes les cellules possibles et renvoie l'objectif le plus proche une fois trouvé

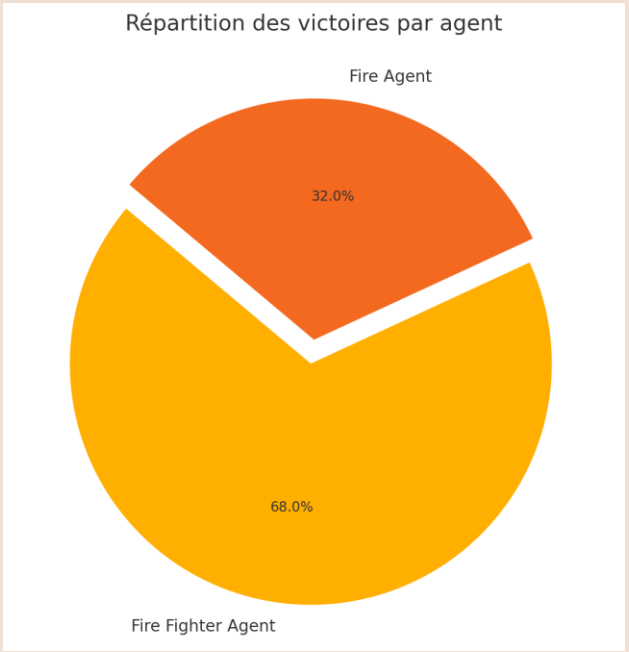


06

Statistiques et Résultats dans le cas d'un environnement statique

Qui est le plus fort ?

Probabilité de gagner dans le cas d'environnement statique pour chaque Agent



68%

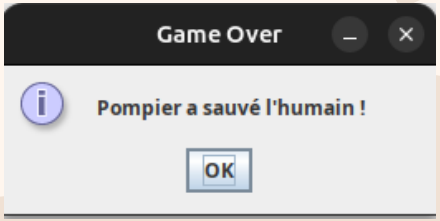
De partie est gagné par le Fire Fighter Agent

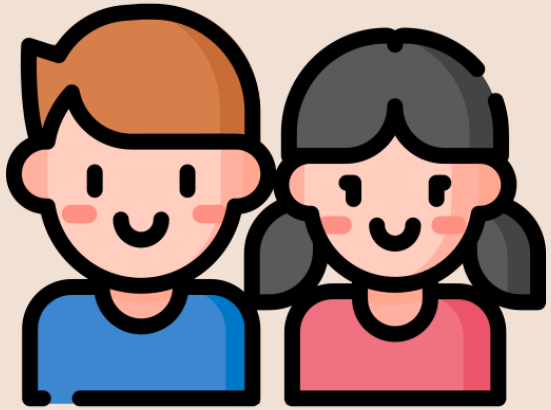
32%

De partie est gagné par le Fire Agent

Pourquoi, ces résultats

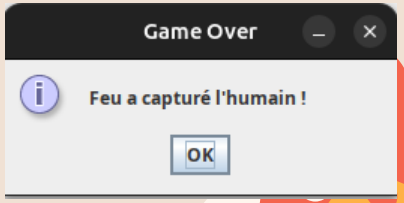
AGENT	Avantages	Inconvénient
FIRE	La possibilité de bruler des objectifs en propageant le feu dans les cases voisines	Inefficacité de l'algorithme BFS dans les environnements statiques et où il y a un seul objectif
FIRE FIGHTER	Mouvement diagonal avec 8 directions (UP, DOWN, LEFT, RIGHT, et diagonales).	Aucun inconvénient notable dans les environnements statique





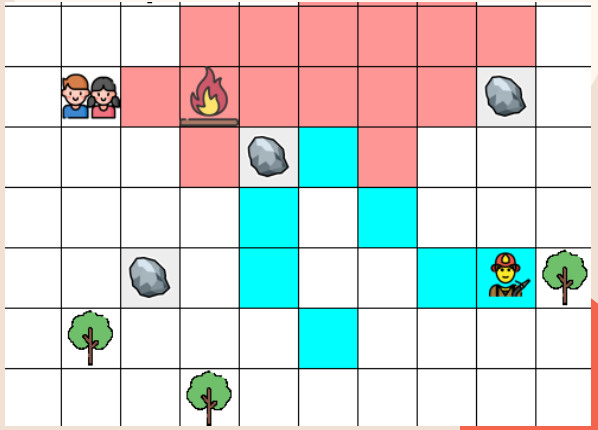
07 l'environnement dynamique!

Release 2.0



l'environnement dynamique!

- **Nouvel Objectif Dynamique : L'Humain**
- **Apparition aléatoire** :
  - Un objectif humain apparaît **tous les 12 à 15 tours**, sur une case vide, ni en feu ni protégée.
- **Règles spéciales** :
  - Le premier agent (Pompier ou Feu) atteignant l'humain déclenche une victoire immédiate.
  - Les 2 agents vont terminer a cherché les objectifs pour déterminer le gagnant en suivant l'ancienne logique
  - Si l'humain n'apparait pas Les 2 agents vont terminer a cherché les objectifs pour déterminer le gagnant en suivant l'ancienne logique
- Dès l'apparition de l'humain :
  - Les agents interrompent leur logique actuelle.
  - Ils recalculent leur déplacement en utilisant la **distance de Manhattan** pour atteindre l'humain.
  - Le jeu se termine dès qu'un agent atteint l'humain.

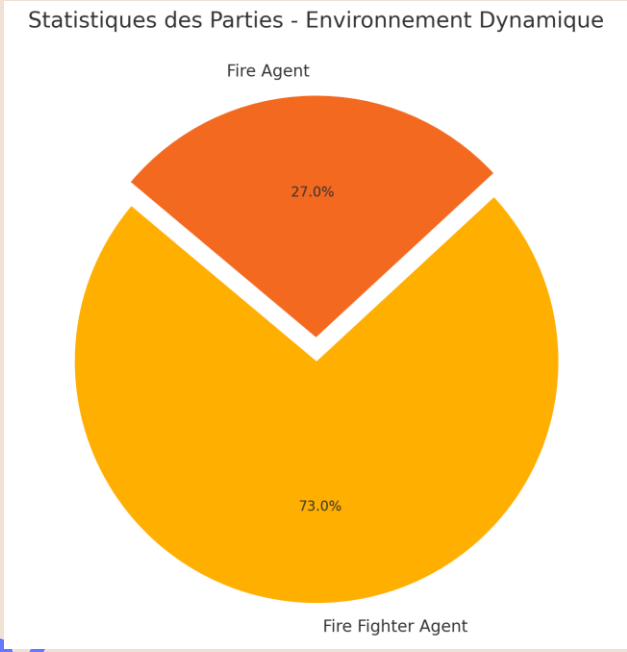


08

Statistiques et Résultats dans le cas d'un environnement dynamique

Qui est le plus fort ?

Probabilité de gagner dans le cas d'environnement dynamique pour chaque Agent



- L’apparition soudaine de l’humain change les priorités des agents et introduit un **élément compétitif décisif**.
- Le jeu devient une **course directe** entre les deux agents pour l’objectif final.
- Dans l'environnement dynamique, le **Fire Fighter Agent** maintient une performance élevée avec 73 % de victoires, bien que légèrement inférieure à l'environnement statique (68 %). L'apparition de l'humain favorise une compétition directe, réduisant l'écart avec le **Fire Agent**.

73%

De partie est gagné par le Fire Fighter Agent

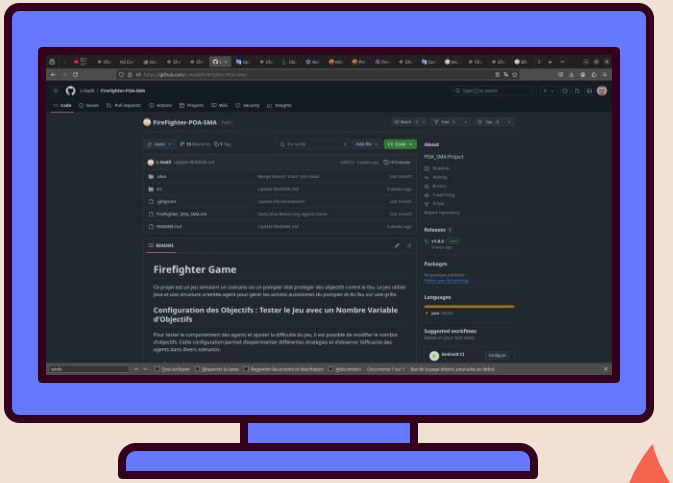
27%

De partie est gagné par le Fire Agent

09

Organisation du Travail

Dévision des taches



Organisation Du Travail

- **Hadil:**
  - Mise en place de l'architecture initiale avec une logique de base pour les agents et l'équilibre du jeu.
  - Développement des règles de score, des conditions de victoire, et des Agents dans un environnement statique.
- **Aziz :**
  - Ajout des icônes et intégration visuelle pour améliorer l'interface utilisateur.
- **Malik:**
  - Optimisation des stratégies des agents avec des algorithmes performants comme Greedy et BFS.
  - Collaboration avec Aziz pour mettre en place l'environnement dynamique.

10

Conclusion

Note sur le projet

notre projet a permis de développer un jeu stratégique avec des agents intelligents, un environnement dynamique et une interface intuitive. Grâce à une répartition équilibrée des tâches et une collaboration efficace, chaque membre a contribué à la réussite et à l'évolution du système.

Thank YOU