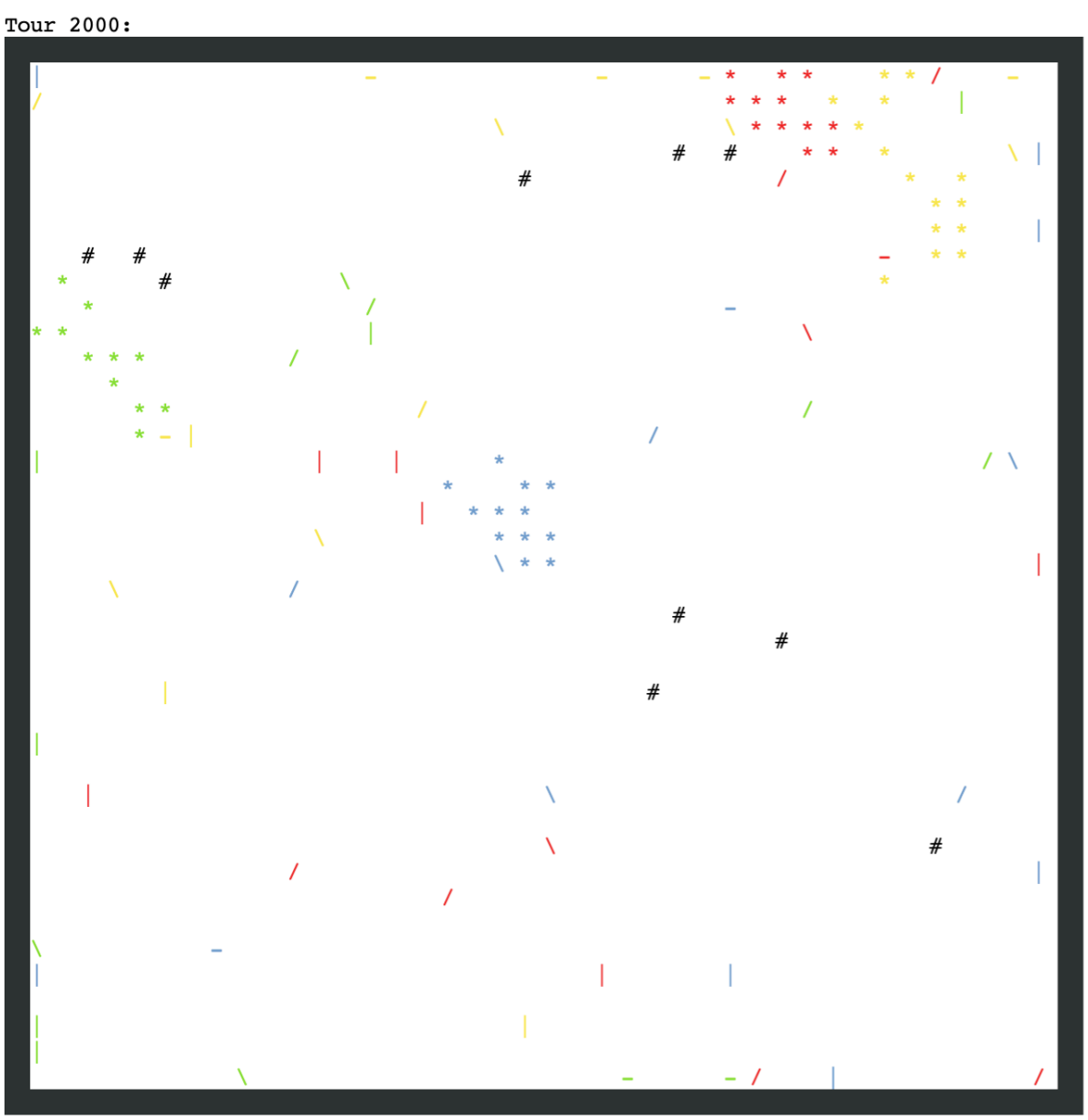
# Description Générale du Projet




---

1. *Utilisation*
2. *Comportement des Termites*
3. *Simulation*
4. *Résultats et Observations*

# 1. Utilisation :

- Le projet comporte 13 fichiers sans compter les exécutables :
    - Les fichiers **monde.cpp**, **termite.cpp**, **grille.cpp** et **coord.cpp**, associés au fichier .**hpp de même nom**, sont le cœur du projet et décrivent toutes les structures et fonctions utilisées.
    - Les fichiers **doctest.h** et **unistd.h** sont des utilitaires décrivant l'infrastructure de test pour le premier, et une commande utile pour la simulation pour le second.
    - Les fichiers **projet.cpp** et **tests.cpp** sont les deux seuls fichiers comprenant un main. Le premier permet de lancer la simulation, le second de lancer tous les tests.
    - Le fichier **Makefile** définit les commandes de compilation du projet.

- Les commandes de compilation sont les suivantes :
    - make : Compile tous les fichiers et crée les deux exécutables projet et tests.
    - make main : Compile tous les fichiers sauf tests.cpp et lance l'exécutable projet.
    - make check : Compile tous les fichiers sauf projet.cpp et lance l'exécutable tests.
    - make all : Compile tous les fichiers et lance les deux exécutables projet et tests.
    - make clean : Efface tous les fichiers .o et les core*

- Plusieurs constantes sont utilisées pour le projet :
    - Ces constantes sont contenues dans le fichier **coord.cpp**.
    - Il y a les **constantes de test** et les **constantes de simulation**. Si vous vous apprêtez à lancer **make check**, utilisez les constantes de **tests**, si vous souhaitez lancer **make main**, utilisez celles de **simulation**. Pour cela, assurez-vous que vos constantes voulues soient visibles par le compilateur et passez les autres en commentaire de la manière suivante :
      */*
      *Constantes*
      **/*
    - Vous pourrez changer à souhait les constantes de simulation (tout en gardant des valeurs respectant les critères décrits à côté de ces constantes), mais ne changez pas les constantes de test.

- Certaines constantes permettent de changer la grille :
    - **tailleGrille** : Défini la taille de la grille, carrée.
    - **nbTermites** : Défini le nombre total de termites, ce nombre sera équitablement divisé entre toutes les équipes.
    - **nbEquipes** : Le nombre d'équipes souhaitées, entre 1 et 4 pour ne pas rendre la simulation trop lourde. On pourrait changer quelques paramètres pour augmenter le nombre d'équipes.
    - **densiteBrindille** : Détermine la densité de brindilles sur la grille.
    - **densiteToile** : Détermine la densité de toiles sur la grille.

- Les autres constantes changent le comportement des termites. Pour conserver l'équité de la simulation, il est impossible de donner des valeurs différentes pour **dureeToile** et **dureeSablier** aux équipes.

## 2. Comportement des Termites :

- On donne à chaque tour la possibilité à tout termite vivant d'effectuer une action :
    - Si le termite est **bloqué** dans une toile d'araignée, rien ne se passe, mais il est 1 tour plus proche d'être libéré. Il sera libéré au bout de **dureeToile tours**.
    - Le termite tente d'abord de **charger une brindille**. Il ne peut charger **qu'une brindille à la fois**, si son **sablier est écoulé** et que cette brindille appartient à une **équipe adverse** ou à **aucune équipe**.
    - Si cela est impossible, il tente d'en **décharger** une. Il ne peut décharger une brindille que si son **sablier est écoulé** et s'il est **en face** d'une brindille de **son équipe** et que cette brindille possède au moins **une case voisine libre**.
    - Si cela est également impossible, on lance la fonction **marcheAleatoire** qui lui proposera 3 actions possibles.
    - Le termite aura une probabilité de **probaTourner** de **tourner d'1/8$^{ème}$ de tour** vers la gauche ou la droite.
    - Sinon, le termite aura une probabilité de **probaCreuser*2** de **creuser un tunnel**. Ce dernier n'est faisable que si le termite ne **porte pas de brindille**, et cette action **téléporte** aléatoirement le termite sur une case vide de la grille.
    - Sinon, le termite **avancera** s'il le peut. Si la case devant lui n'est pas libre, il **tournera d'1/8$^{ème}$ de tour**.

- Afin de garder la grille et le vecteur de termites cohérents, on utilise une structure Monde et différentes méthodes lors de l'appel des fonctions :
    - La structure **Monde** contient une **grille** et un **vecteur**. On récupère dans **projet.cpp** un **pointeur** vers ces deux attributs afin de lancer la simulation.
    - Quand on agit sur un termite, la structure est automatiquement modifiée pour ce termite.
    - On passe la grille par **référence** aux **méthodes de la structure Termite** afin d'y effectuer les mêmes modifications que sur le termite.

## 3. Simulation :

- Une simulation dure **2 000 tours** et donne vie à chaque termite une fois par tour. On veille constamment à l'intégrité de la simulation avec les fonctions prévues à cet effet. On utilise 3 modes différents afin de simuler le monde :
    - Le **mode 1** permet de lancer une unique simulation sur une grille aléatoire. On joue au **tour par tour** et l'utilisateur appuie sur Entrée pour passer au tour suivant (ce qui affiche la grille dans l'état actuel du jeu). L'utilisateur peut également appuyer sur Espace puis Entrée afin **d'afficher l'état du vecteur de termites**. Enfin, il peut appuyer sur une touche quelconque puis Entrée pour quitter la simulation. A la fin, le programme affiche le nombre de brindilles par équipe.
    - Le **mode 2** est le même que le mode 1 mais les tours sont passés **automatiquement** toutes les 0,015 secondes, suffisant pour voir proprement la grille. Il n'y a aucune intervention nécessaire de l'utilisateur et ce dernier ne pourra pas afficher le vecteur de termites, sauf une fois la simulation finie et les résultats affichés, où une option lui sera proposé, comme dans le mode 1, pour **afficher le vecteur** en appuyant sur Espace puis Entrée. Il appuiera sur une touche quelconque puis Entrée pour terminer et récupérer la main. Ce mode prend **30 secondes**.
    - Le **mode 3** joue **600 simulations** et affiche le nombre **total de brindilles par équipe** lors de ces dernières. Cela permet d'influer sur les constantes **probaCreuser** et **probaTourner** des équipes et de déterminer les constantes les plus **efficaces**. Ce mode peut prendre jusqu'à **10 minutes**.

- Une fonction tuera aléatoirement des termites, jusqu'à **5%** par partie (arrondi à l'entier inférieur). Les termites seront tués à des tours précis afin de tuer **exactement un termite** pour un **nombre de tours** calculé en fonction du nombre de termites. Les termites morts seront **sortis de la grille et inactifs**.

## 4. Résultats et Observations :

- On a donné aux 4 équipes les constantes suivantes :

| Equipes : | probaTourner | probaCreuser |
|---|---|---|
| **A** | 0.1 | 0.05 (=0.0025) |
| **B** | 0.2 | 0.05 (=0.0025) |
| **C** | 0.1 | 0.09 (=0.0081) |
| **D** | 0.2 | 0.09 (=0.0081) |

- On a joué plusieurs fois le mode 3 et récolté les résultats, voici quelques exemples des résultats obtenus :

```
Affichage des résultats:
Equipe 0: 6172
Equipe 1: 5134
Equipe 2: 6515
Equipe 3: 5198
```

```
Affichage des résultats:
Equipe 0: 6776
Equipe 1: 4955
Equipe 2: 6189
Equipe 3: 5199
```

- En ayant joué tant de simulations, on peut s'affranchir de l'aléatoire du placement des nids au début et du mouvement des termites et conclure sur les constantes :
    - Plus la constante **probaTourner** est faible, plus les termites sont **efficaces**. Ceci est logique puisque moins les termites tournent, plus ils ont de chance de parcourir toute la grille.
    - La constante **probaCreuser** ne change visiblement **rien** et ne permet pas à une équipe d'être meilleure que l'autre.