



# Abeilles versus frelons

## Table des matières

1	L'objectif du projet	1
2	Implantation et niveaux de jeu	4
3	Améliorations	9
4	Travail et rendu	9
5	Derniers conseils	10

## 1 L'objectif du projet

Le but du projet est de simuler de manière (très) simplifiée les interactions entre des abeilles et des frelons. Le projet repose sur l'utilisation de listes chaînées et la bibliothèque graphique de l'université, la `libMLV`. Il est à réaliser en **binôme** (ou exceptionnellement seul.e, contactez vos enseignants dans ce cas) et à développer par étapes successives.

Deux camps sont présents, les abeilles et les frelons. La partie se déroule sur une grille rectangulaire. Les abeilles sont structurées en cinq types :

- les **ruches** qui ne se déplacent pas. Elles peuvent produire des reines, des guerrières, des ouvrières, et des escadrons de guerrières,
- les **reines**, qui se déplacent, attaquent ou construisent une ruche,
- les **guerrières**, qui se déplacent et attaquent,
- les **escadrons** de guerrières, qui se déplacent et attaquent,
- les **ouvrières**, qui se déplacent ou récoltent du pollen.

Le pollen récolté constitue la ressource des abeilles pour leur développement. Il est partagé par l'ensemble des ruches. Les frelons sont structurés en trois catégories :

- les **nids** qui ne se déplacent pas. Ils peuvent produire des reines, et des frelons,
- les **reines**, qui se déplacent, attaquent ou construisent un nid,
- les **frelons**, qui se déplacent et peuvent attaquer.

Les ressources des frelons sont les abeilles défaits au combat. On appellera cette ressource la *ressource abeille*. Elle est partagée par l'ensemble des nids de frelons.

On appellera **unité** une ruche, un nid, une reine, une ouvrière, une guerrière, un escadron ou un frelon.

### 1.1 Mise en place de la simulation

La simulation se fera sur une grille rectangulaire de 18 lignes et 12 colonnes, et on note  $(i, j)$  la case située à la  $i$ -ième ligne et la  $j$ -ième colonne, de sorte que  $(0, 0)$  se situe en haut à gauche. On considère que chaque case a **8 voisins**. Les bords du plateau ne sont pas franchissables. Initialement, il y a une ruche, une reine, une ouvrière et une guerrière côté abeilles ; côté frelons, il y a un nid, une reine et deux frelons. La ruche et le nid sont placés dans des coins opposés, en  $(0, 0)$  et  $(18, 12)$ . On

pourra initialiser les ressources à 10 unités de pollen pour les abeilles et 10 ressources abeille pour les frelons.



Au début de chaque tour, chaque unité se situe sur une case. Les insectes pourront se déplacer, et combattre lorsqu'ils se trouvent sur la même case après les déplacements de chacun. Un tour se déroule de la manière suivante :



- Au début du tour, on tire au hasard qui parmi les abeilles ou les frelons agira en premier.
- Le premier camp donne d'abord les commandes de production des ruches ou des nids, puis les ordres à ses insectes. C'est ensuite le tour du deuxième camp.
- Après cela seulement, les combats sont résolus (voir plus bas pour la résolution des combats).
- Les ressources des deux camps sont ensuite mises à jour. Les ouvrières qui ont récolté ce tour sans être détruites ajoutent 1 unité de pollen chacune, tandis que les abeilles détruites lors des combats donnent des points de ressource abeille aux frelons.
- Le tour est terminé.

## 1.2 Ruches et nids

Les insectes sont produits par les ruches des abeilles et les nids des frelons. Les coûts de production des abeilles sont mesurés en unité de pollen (récolté par les ouvrières) et ceux des frelons sont mesurés en nombre d'abeilles (obtenues lors des combats). Une reine abeille ou frelon peut fonder une ruche ou un nid en 1 tour pour respectivement 10 unités de pollen et 10 abeilles.

Les ruches et les nids peuvent produire des insectes. Une fois la production d'une ruche ou d'un nid choisie, elle ne peut être changée jusqu'à sa réalisation complète. Voici les temps en nombre de tours, et les coûts de production pour chaque camp. Ces valeurs par défaut doivent pouvoir facilement être changées.

<b>Abeilles</b> 	Coût 	Temps
Reine	7	8
Ouvrière	3	2
Guerrière	5	4
Escadron	6	6

<b>Frelons</b> 	Coût 	Temps
Reine	8	8
Frelon	3	5

Après avoir terminé, la ruche ou le nid attend un nouvel ordre de production.

Abeilles et frelons ont un système d'**affiliation par ruche et par nid**. Tous les membres produits par une ruche ou par un nid sont affiliés à cette ruche ou à ce nid. De plus, lorsqu'une reine abeille ou frelon construit une ruche ou un nid, elle se rattache alors à celui-ci.

## 1.3 Choix des actions

Pour repérer aisément une unité en attente d'ordre, on l'affiche avec une couleur différente.

### 1.3.1 Reines

Les reines abeilles et frelons ont la possibilité de

1. Se déplacer vers une case voisine
2. Fonder une ruche ou un nid (un seul par case, 1 tour)
3. Passer le tour

Une reine ne peut fonder qu'une seule nouvelle ruche ou nid.

### 1.3.2 Guerrières, escadrons et frelons

Ces trois classes d'insectes ont la possibilité de

1. Se déplacer vers une case voisine
2. Passer le tour

### 1.3.3 Ouvrières

Une ouvrière a la possibilité de

1. Se déplacer vers une case voisine
2. Récolter du pollen (4 tours)

Le pollen peut être récolté sur n'importe quelle case de la grille qui n'est pas occupée par une unité des frelons. Pour effectuer la récolte, l'ouvrière reste immobilisée pendant 4 tours, après quoi elle meurt (l'unité est détruite, sans rapporter de ressource abeille aux frelons). A chaque tour de récolte et au moment de la mise à jour des ressources, 1 unité de pollen est collectée et s'ajoute au compte de ressources des abeilles.

## 1.4 Combat

Lorsque 2 unités adverses se retrouvent sur la même case à la fin d'un tour, un combat a lieu. L'issue du combat est déterminée par tirage aléatoire d'un dé à 60 faces (fonction `int MLV_integer_random(int begin, int end)` par exemple). Un coefficient égal à leur force respective sera appliqué. Par exemple, si une abeille guerrière et un frelon combattent, on compare `tireDe() x FGUERRIERE` et `tireDe() x FFRELON` et le plus grand score l'emporte.

À l'issue d'un combat, les règles suivantes s'appliquent :

- Lorsqu'un insecte perd un combat, il est détruit.
- Lorsqu'une reine frelon ou un frelon l'emporte sur une ruche, **la ruche est transformée en nid de frelons**, et toutes les abeilles affiliées sont détruites. Le frelon ayant vaincu la ruche se retrouve affilié à ce nouveau nid.
- Toutes les abeilles détruites lors d'un combat ou la capture d'une ruche apportent des points de ressources abeille pour les frelons. Le nombre de ressources abeilles gagnées selon qui est vaincu ou détruit correspond au nombre d'unités de pollen qui ont été nécessaires à leur production initiale, soit

Abeille détruite	Ressource abeille rapportée
Reine	7
Ouvrière	3
Guerrière	5
Escadron	6

Lorsque plusieurs unités d'un même camp sont sur une même case, le processus se répète tant qu'un des camps n'est pas éliminé de la case. On ordonne les unités occupant une même case pour qu'interviennent en premier :

- les escadrons, les guerrières, les reines et enfin les ouvrières côté abeilles ;
- les frelons, puis les reines côté frelons.

Si une ruche ou un nid est présent, il n'est attaqué que lorsqu'aucun insecte de son camp n'est présent pour le défendre.

## 1.5 Fin de partie

Une partie est terminée lorsque toutes les ruches d'abeilles ou tous les nids de frelons ont été détruits. On peut également choisir de quitter la partie.

## 1.6 Sauvegarde et chargement

Au début du tour, après le tirage aléatoire décidant qui commence le tour, le camp qui joue en premier peut choisir de sauvegarder la partie dans un fichier. Le format de sauvegarde est décrit en Section 2.4 plus bas. Il "photographie" la partie à l'instant de la sauvegarde en enregistrant quel camp doit agir, ses ressources et unités et celles de l'adversaire.

On devra de même pouvoir choisir de récupérer une partie mémorisée dans un fichier. Dans ce cas, l'utilisateur entrera le nom du fichier de sauvegarde.

## 2 Implantation et niveaux de jeu

Le projet finalisé utilise une interface graphique obtenue via la `libMLV`. Un niveau moins abouti du projet peut se contenter d'une interface sur terminal.

Le jeu se joue au tour par tour. À la fin de son tour, le joueur doit indiquer sa fin de tour afin de rendre la main à son adversaire.

## 2.1 Interface terminal

Afin de mettre en place les premières étapes de la simulation plus rapidement, on peut commencer en faisant l'économie de l'interface via la `libMLV` et en utilisant le terminal à la place. Par exemple (avec une grille plus petite pour améliorer la visibilité) :

R go - r														
														-Nrf

Tour des abeilles  
Pollen : 45  
Choix de la production de la ruche en (0,0)  
1. Produire une reine (20 pollen, 5 tours)  
2. Produire une ouvriere (20 pollen, 5 tours)  
3. Produire une guerriere (20 pollen, 5 tours)  
4. Produire un escadron (20 pollen, 5 tours)  
5. Passer le tour  
6. Detruire l'unite

Dans une case, on indiquera les unités qui peuvent s'y trouver par le code suivant : **Rroge-Nrf**. Les 5 premiers symboles correspondent aux abeilles : **R** si une ruche est présente, **r** si une reine abeille est présente, **o**, **g**, **e** si une ouvrière, une guerrière et un escadron est présent. Les 4 dernières lettres correspondent aux frelons : **N** si un nid est présent, **r** si une reine frelon est présente et **f** si un frelon est présent. En cas d'absence, on ne met pas la lettre et on laisse un espace. En particulier, il ne peut y avoir une ruche et un nid présents sur la même case. Par exemple, **R go - r** signifie qu'il y a une ruche, une guerrière, une ouvrière et une reine frelon sur la case.

Pour les déplacements des insectes, on demandera au joueur d'entrer la direction de la case voisine où il veut se déplacer : **N** (nord), **NE** (nord-est), **E** (est), **SE** (sud-est), **S** (sud), **SO** (sud-ouest), **O** (ouest), **NO** (nord-ouest).

## 2.2 Interface graphique

L'interface graphique doit respecter les impératifs suivants :

- une partie représente la grille de jeu avec son quadrillage apparent

- chaque case est de taille 60 par 60 pixels et se découpe elle-même en 7 zones de la façon suivante

	Reine abeille	Reine frelon
Ouvrière	Ruche ou Nid	Frelon
	Guerrière	Escadron

On pourra représenter la ruche / le nid et les insectes par des carrés pleins de couleurs différentes (voire utiliser des images de `man MLV_image.h`) ;

- la seconde partie forme une barre de menu contenant des boutons permettant la gestion des différents ordres à transmettre aux ouvrières, guerrières, escadrons, frelons, reines, ruches et nids ; elle comprend également une partie dans laquelle s’afficheront les coordonnées de l’unité en cours de traitement, et l’action qu’elle est en train d’effectuer ;
- enfin une dernière zone contient un bouton **sauve**, un bouton **charge**, un bouton **quitter**, un bouton **fin de tour** ainsi que l’indication du numéro du tour et la clan du joueur en cours.

Pour déplacer un insecte sur une case voisine, on cliquera sur la case au moment du tour de cet insecte.

## 2.3 Structures et représentations des données

Les ruches, nids, et insectes sont tous représentés par la même structure `Unite` (pour “unité”). En fonction des étapes d’implémentations réalisées, certains champs peuvent ne pas être utilisés. On utilisera les constantes symboliques et les structures suivantes :

```
// Dimensions de la grille en nombre de cases (origine en haut a gauche) :
#define COLONNES 12
#define LIGNES 18

// Les deux camps :
#define ABEILLE 'A'
#define FRELON 'F'

// Les types d'unites :
#define REINE 'r'
#define OUVRIERE 'o'
#define ESCADRON 'e'
#define GUERRIERE 'g'
#define FRELON 'f'
#define RUCHE 'R'
#define NID 'N'
// Pour la recolte de pollen
#define RECOLTE 'p'

// Les temps necessaires a la production :
#define TREINEA 8
#define TREINEF 8
#define TOUVRIERE 2
#define TGUERRIERE 4
#define TESCADRON 6
#define TFRELON 5
#define TRECOLTE 4
```

```

// Les couts necessaires a la production :
#define CREINEA 7
#define CREINEF 8
#define COUVRIERE 3
#define CGUERRIERE 5
#define CESCADRON 6
#define CFRELON 3
#define CRUCHE 10
#define CNID 10

// La force des unites
#define FREINE 6
#define FOUVRIERE 1
#define FGUERRIERE 5
#define FESCADRON 12
#define FFRELON 8

// La structure Unite :
typedef struct unite {
    char camp; // ABEILLE ou FRELON
    char type; // RUCHE, NID, REINE, OUVRIERE, GUERRIERE, ESCADRON ou FRELON
    int force; // la force de l'unite
    int posx, posy; // position actuelle sur la grille
    int destx, desty; // destination (negatif si immobile)
    char production; // production d'une ruche ou d'un nid et RECOLTE pour la recolte de pollen
    int temps; // nombres de tours total pour cette production
    int toursrestant; // tours restant pour cette production
    struct unite *usuiv, *uprec; // liste des unites affiliees a une ruche ou un nid
    struct unite *colsuiv, *colprec; // liste des autres ruches ou nids (colonies) du même camp
    struct unite *vsuiv, *vprec; // liste des autres unites sur la meme case
} Unite, *UListe;

// La structure Case :
typedef struct {
    Unite *colonie; // S'il y a une ruche ou un nid sur la case
    UListe occupant; // les autres occupants de la case
} Case;

// La structure Grille :
typedef struct {
    Case plateau[X][Y];
    UListe abeille, frelon;
    int tour; // Numero du tour
    int ressourcesAbeille, ressourcesFrelon;
} Grille;

```

On rappelle que les insectes produits par une ruche ou un nid sont affiliés à celui-ci. De plus, une reine abeille ou frelon, fondatrice d'une ruche ou d'un nide, se retrouve affiliée à celui-ci.

## 2.4 Format de sauvegarde

Le format des fichiers de sauvegarde est fixé ainsi. La première ligne contient un **char** et deux **int** séparés par un espace : la lettre ('A' ou 'F') précisant le camp devant jouer et ses ressources (en unité de pollen pour les abeilles, et en abeilles pour les frelons), puis les ressources de l'autre camp.

Sur chacune des lignes suivantes, on trouve les caractéristiques d'une seule unité (sauf bien entendu les valeurs des pointeurs). Une ligne est donc constituée de 2 **char** suivis de 2 **int**, puis d'1 **char** et enfin d'1 **int** séparés par un seul espace.

Ce formatage doit permettre d'échanger des fichiers de tests.

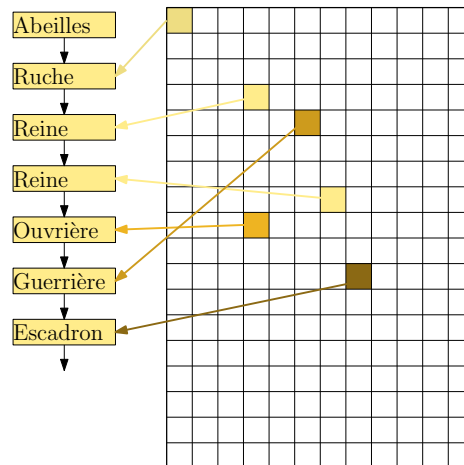
## 2.5 Les étapes de développements

On propose aux étudiants de développer le projet par étape de difficultés croissantes, et qui correspondent à des notes maximales croissantes.

### 2.5.1 Etape "Colonie d'abeilles"

On demande d'abord de gérer les déplacements des abeilles et la production d'une unique ruche. On doit pouvoir choisir la production de la ruche, déplacer ou détruire un insecte et récolter pour les ouvrières. À chaque tour, il faut afficher la liste ordonnée (reines, escadrons, guerrières, puis ouvrières pour les abeilles) de toutes les unités présentes.

Le schéma de pointage est le suivant.



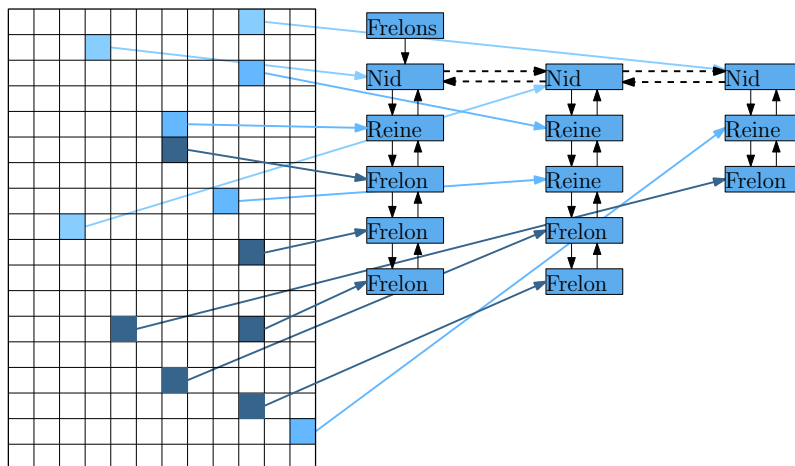
A cette étape, deux unités ne peuvent pas être placées sur la même case. Un insecte produit par une ruche est placé sur une case libre voisine. Une case pointe sur l'unité située sur cette case.

### 2.5.2 Etape "Nid de frelons"

Même travail que dans l'étape "Colonie d'abeilles", mais pour ajouter un nid de frelons et ses reines et frelons affiliés. Le schéma de pointage est le suivant.







### 2.5.5 Etape "Sauvegarde"

On ajoute dans cette étape la gestion des sauvegardes et chargements de parties.

### 2.5.6 Etape "Interface graphique"

L'utilisation d'une interface graphique est obligatoire à cette étape.

### 2.5.7 Etape finale

On autorise désormais la possibilité que plusieurs unités soit placées sur la même case. Pour cela, il faudra utiliser la liste doublement chaînée des unités situés sur la même case (`*vsuiv`, `*vprec`). On pourra ordonner les unités dans la chaîne selon l'ordre de participation aux combats (voir Section 1.4).

## 3 Améliorations

Des améliorations sont envisageables mais ne seront prises en compte que si toutes les étapes précédentes sont réalisées. On peut par exemple :

- Ajouter un système de points de vie et de points d'attaque à toutes les unités.
- Autoriser des déplacements de 2 cases aux ouvrières sauf si elles sont adjacentes à une unité du camp frelon
- Relier la force d'une ruche aux nombre de guerrières et d'escadrons affiliés. Relier la force d'un nid au nombre de frelons affiliés.
- Relier les temps de production d'une ruche au nombre d'ouvrières.

## 4 Travail et rendu

Le projet est à effectuer en **binôme**. Le rendu comprendra :

- les programmes C compilables à l'université ;
- un rapport décrivant :
  - d'une part, le fonctionnement du jeu ;
  - d'autre part, les méthodes utilisées et les choix d'implantation.

Le but du rapport est de permettre à un utilisateur quelconque d'utiliser le programme, et à un programmeur averti de faire évoluer facilement votre code.

Une soutenance aura lieu à la rentrée en janvier 2024. La compilation de votre programme, ainsi qu’une démonstration de votre code sur une machine de l’université, fait partie de la soutenance. Ensuite, une série de questions sur votre travail sera posée à chaque membre du binôme.

## 5 Derniers conseils

Les programmes `C` devront être clairs et judicieusement commentés. Ne commencez pas à développer une étape avant que la précédente ne soit pas totalement fonctionnelle.

Sauvegardez, étape par étape, des versions fonctionnelles pour pouvoir repartir sur de bonnes bases lorsque de nouvelles fonctionnalités se révèlent trop difficiles à implanter. Chacune des versions pourra être testée le jour de la soutenance.