CY Cergy Paris Université

Département Science Informatique

GUIDE PROGRAMMEUR

Simulation d'un tournoi sportif

Rédigé par :

- ABDERRAHMANE Saber
- MESBAHI Said
- KESKES Mohamed Nazim

3 avril 2023

Introduction :

Le thème du programme réalisé est la simulation d'un tournoi sportif à élimination directe. L'objectif de ce dernier est de permettre aux utilisateurs de simuler facilement des tournois de différentes tailles et avec différents paramètres tels que le nombre d'équipes et les règles de qualification. Le programme est conçu pour être flexible et personnalisable afin de répondre aux besoins de divers types de tournois.
Le programme permet également d'exporter les résultats du tournoi sous forme de fichier csv ou txt pour une analyse ultérieure.
Le but de ce guide du programmeur est donc de fournir une documentation détaillée sur la manière d'utiliser et de modifier le code source pour personnaliser le programme selon les besoins de chaque utilisateur.

Analyse du Projet :

- **Identification des règles du tournoi à simuler :**

Pour identifier les règles du tournoi à simuler, nous avons étudié les règles en vigueur dans les principaux tournois sportif et avons élaboré un ensemble de règles génériques pour permettre une simulation "réaliste" et flexible, comme la limitation de la durée d'un match ou encore le nombre d'équipe participante devant être une puissance de 2.
Les règles ont été formalisées dans une spécification détaillée qui a servi de base pour le développement de la simulation.
Il est également prévu des options de configuration pour permettre aux utilisateurs de personnaliser le score des équipes en fonction de leurs besoins spécifiques.

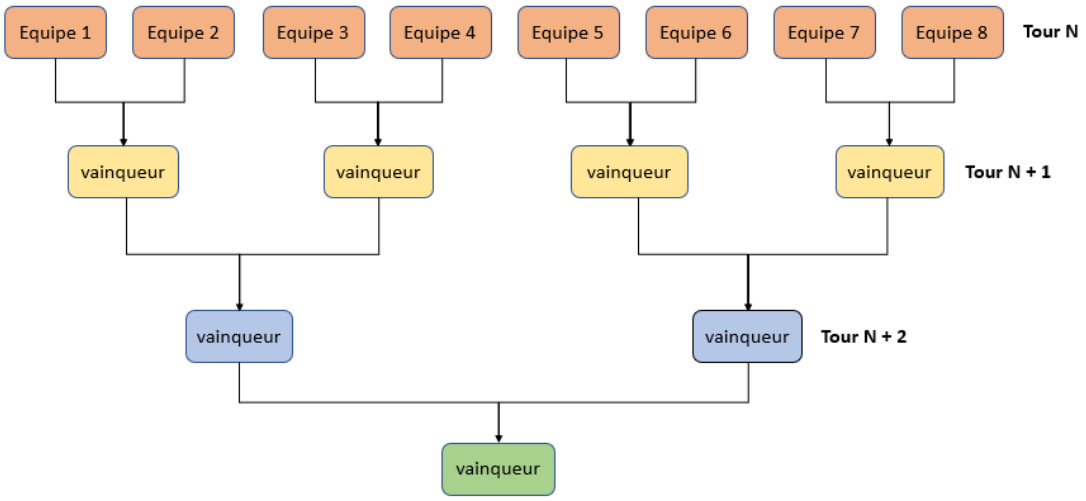

Figure 1 : Prototype de tournois sur lequel est basé le programme.

- **Identification de l'implémentation des fonctionnalités :**

Pour implémenter les fonctionnalités du programme de simulation de tournoi, il est question d'utiliser des processus grâce à la fonction fork() pour gérer la concurrence entre les différentes tâches. Nous avons également utilisé des sémaphores pour assurer la synchronisation entre les processus et éviter les conflits de ressources.
Plus précisément, nous avons utilisé la fonction fork() pour créer un processus fils pour chaque match à jouer dans le tournoi. Les processus fils sont ensuite synchronisés à l'aide

de sémaphores pour éviter les conflits de ressources et garantir que les résultats du match sont correctement enregistrés.
Nous avons également utilisé des sémaphores pour éviter que plusieurs processus ne tentent d'écrire dans le même fichier de résultats en même temps.

- **Conception de l'architecture du programme :**

La conception de l'architecture du programme a été basée sur une approche modulaire, permettant une meilleure lisibilité et maintenabilité du code. Le programme est divisé en plusieurs modules,chacun gérant une partie spécifique de la logique du tournoi.
Les modules sont les suivants :

- **le fichier "ipcTools.h"** : contient la déclaration de fonctions pour créer,obtenir et libérer des sémaphores et des segments de mémoire partagée.
- **Le fichier principal** qui contient la fonction main() ainsi que toutes les fonctions implémentées pour le programme :
    - **simule() :** cette fonction simule un match entre deux équipes et met à jour leur état (qualifié ou non) en fonction du score.
    - **simule_man()):** cette fonction permet à un utilisateur de saisir le score d'un match. (extension).
    - **cleanup() :** cette fonction libère tous les sémaphores créés et détruit le segment de mémoire partagée.

Réalisation du projet :

- **<u>Description de l'environnement de développement :</u>**

L'environnement de développement utilisé pour la création du programme est le suivant :

- **Système d'exploitation :** UNIX
- **Éditeur de texte :** Visual Studio Code
- **Langage de programmation :** C
- **Outils de compilation :** GCC, make
- **Outils de débogage :** GDB
- **Gestionnaire de version :** Git

Nous avons également utilisé des bibliothèques standard du langage C, telles que stdio.h, stdlib.h, string.h, ainsi que des bibliothèques spécifiques à Unix comme sys/wait.h ou encore semaphores.h.

- **<u>Comment on a pu implémenter les fonctionnalités principales (gestion des équipes, matchs et la synchronisation des tours) :</u>**

Le programme utilise des sémaphores pour gérer l'accès concurrent à une structure de données partagée contenant les informations sur les équipes ainsi que de leur statut lors de la compétition.
L'utilisation des sémaphores permet de gérer l'accès concurrent à la structure de données partagées, en créant un certain nombre de sémaphores pour chaque tour de la compétition, où chaque équipe est associée à deux sémaphores : un pour l'accès en lecture et un pour l'accès en écriture. Le programme utilise des opérations P et V sur ces sémaphores pour synchroniser l'accès concurrent à la structure de données partagée.

Il est également question d'utiliser un sémaphore mutex pour synchroniser l'accès concurrent aux variables du programme qui stockent les résultats du match et le statut des équipes, ceci garantit que deux processus ne modifient pas ces variables simultanément.

Déroulement :

Le programme se déroule en plusieurs étapes. Tout d'abord, il lit les données du fichier d'entrée qui contient les informations sur les équipes participant au tournoi. Ensuite, le programme place les équipes d'une façon randomisé dans un tableau en leur attribuant un statut, ce statut est initialisé à 1 au début et peut prendre plusieurs valeurs (0 , 1) sachant que :

0 : Équipe non qualifiée.
1 : Équipe qualifiée du tour

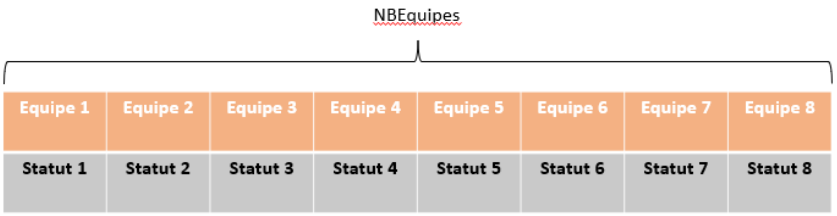

Figure 2 : Le tableau Equipe, utilisé afin de simuler le tournoi

La simulation de matchs utilise des sémaphores pour synchroniser l'accès aux données partagées et éviter les conflits de lecture/écriture. Les sémaphores sont initialisés avant l'exécution de la fonction, et sont utilisés à l'aide des fonctions "P" et "V" selon le besoin. Au départ, il est nécessaire d'initialiser en utilisant le PID du processus courant afin d'obtenir des résultats différents à chaque exécution de la fonction.

Ensuite l'indice k est calculé à partir de l'identifiant "id" en utilisant "int k = id - 1". Cet indice est utilisé pour déterminer les sémaphores correspondant au processus courant dans la matrice "tabSem".

Les sémaphores sont ensuite utilisés pour protéger les ressources partagées (ici le tableau de sémaphores) entre différents processus.

Une fois que les sémaphores correspondant au processus courant sont acquis, le programme parcourt la zone de mémoire partagée "tab" à la recherche des indices i1 et i2 qui représentent les indices des deux premiers matchs disponibles pour le processus "id", sachant que pour trouver i1, une formule mathématique est utilisée :

int i1=((int)pow(2, t))*k;

sachant que, "t" représente le numéro du tours en cours et "k" l'indice de l'équipe courante, la fonction pow(2,t) donne le nombre de matchs joués lors des tours précédents.

int i2 = i1 + 1

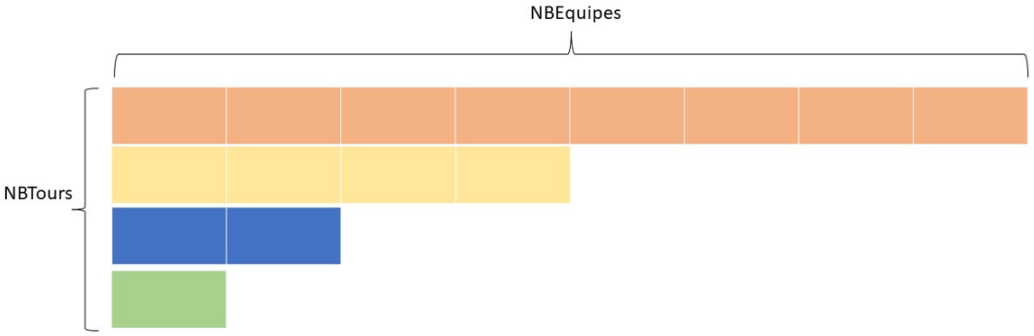

Figure 3 : Le tableau de sémaphore utilisé pour la synchronisation des tours

La partie suivante du code génère les résultats aléatoires des matchs et met à jour le statut des matchs dans le tableau des matchs partagés.
La fonction affiche les résultats du match avec le nom des équipes, le score ainsi que les actions. Enfin, la fonction V est appelée pour déverrouiller le sémaphore “mutMatch” afin que d’autres processus puissent accéder à la section critique.

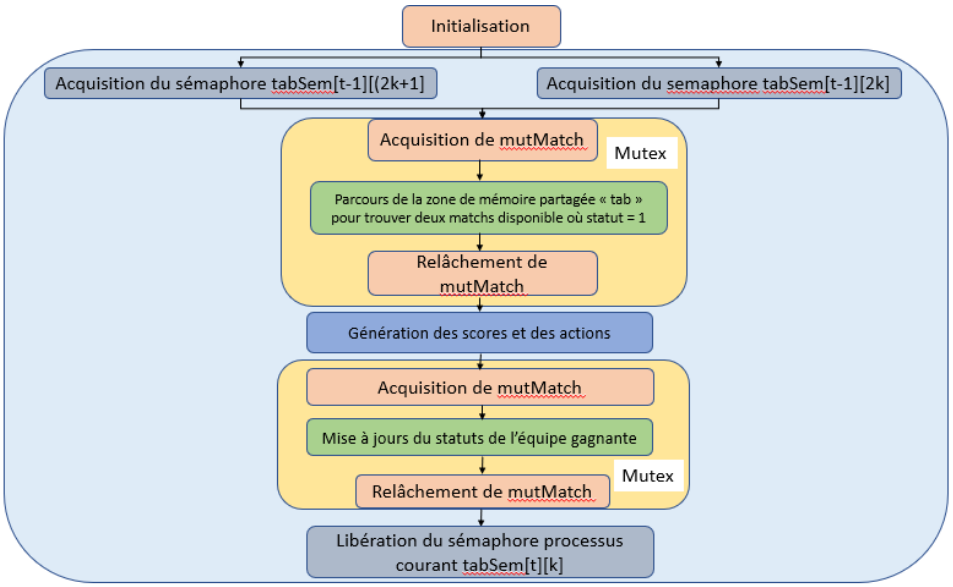

Figure 4 : Concept de la simulation de match

Après que tous les matchs ont été joués , les équipes avec le statut à 1 dans chaque groupe sont qualifiées pour la phase suivante. en cas d'égalité de score. Il choisit une équipe de manière aléatoire en utilisant une fonction de randomisation. Dans la phase suivante, le programme simule des matchs à élimination directe entre les équipes qualifiées jusqu'à ce qu'un vainqueur soit déterminé.

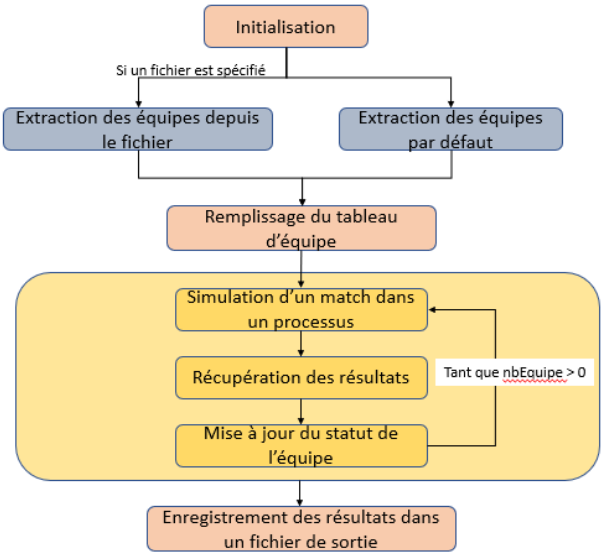

Figure 5 : Principal prototype sur lequel est basé le programme.

Extension : Ajout d'un mode manuel

La fonction "simule_man" permet de simuler un match entre deux équipes , en prenant en entrée l'identifiant du match et le numéro du tour.
La fonction utilise la bibliothèque "srandom" pour initialiser la génération de nombres aléatoires, puis utilise des sémaphores pour synchroniser l'exécution du match et pour eviter que deux équipes ne disputent un match en même temps.
Cette fonction permet spécialement de laisser à l'utilisateur la saisie du score de chaque équipe, simule un temps d'attente aléatoire, puis met à jour le statut des équipes pour indiquer si l'équipe gagnante a avancé au tour suivant.
Finalement, la fonction affiche le résultat du match, y compris le score et l'identifiant du match, ainsi que le numéro du tour. Elle utilise également des sémaphores pour signaler la fin du match.

Conclusion :

- **Bilan :**

- Le programme de simulation de tournois a été conçu et développé avec succès.
- Les fonctionnalités principales du programme ont été implémentées, telles que la génération aléatoire des résultats, le calcul des scores, la gestion des éliminations, etc.
- L'implémentation avec succès des fork et des sémaphores pour assurer la gestion de l'exécution des tâches en parallèle et éviter les problèmes de concurrence.

- **Limitation et perspectives d’améliorations :**

  Limitation :
- Le programme est actuellement limité à un tournoi à élimination simple et ne prend pas en charge les tournois à double élimination ou les phases de groupe.
- Le programme en mode console est spécifique à Unix et ne fonctionne pas sur Windows.

  Perspectives d’amélioration :
- Ajouter la prise en charge de différents types de tournois, et laisser le choix à l’utilisateur de choisir son tournoi au début du programme.
- Développer une interface graphique utilisateur pour rendre le programme plus convivial.
- Proposer une version du programme en mode console compatible avec Windows.

- **Répartition des tâches pour la réalisation de ce projet :**

Pourcentage d'implications des membres du groupe :

NOM Prénom	**Tâches réalisées**	**Pourcentage d'implication**
KESKES Nazim	-Gérer la génération aléatoire des scores et leur affichage en temps réel. -La mise en place du fichier de sortie pour le déroulement du tournoi (csv,txt) -Optimisation de la gestion des ressources pour assurer de bonnes performances pour le programme	**33.33%**
MESBAHI Said	-Conception et développement de la structure de base du programme. -Gérer la lecture des données depuis le fichier d'entrée -Rédaction de la documentation pour les développeurs.	**33.33%**
ABDERRAHMANE Saber	-Mise en place de la synchronisation des matches et la gestion des équipes -Mise en place du système de configuration pour les paramètres tels que la durée des matchs. -Rédaction du guide utilisateur - Production de la documentation Doxygen pour le code source	**33.33%**