# Guide du développeur

sur le sujet

# Logiciel de simulation de tournoi sportif

rédigé par

**Linxiang CONG, Yicheng QIAN**

# Table des matières

# 1 Introduction

## 1.1 Introduction du projet

Il s'agit d'un logiciel de simulation de tournoi sportif (football, rugby ou autre), écrit en C sous le système de Linux.

## 1.2 Introduction à la structure de donnée

Dans le dossier du projet, on a utilisé une structure la plus simple : tableau. Compte tenu des manipulations dans la mémoire partagée, Utilisation du tableau me permet de faciliter la tâche. Et ce tableau est de type 'Team'. Pour la structure 'Team', elle comme suivant :

```
/**
 * Team struct
 */
typedef struct team{
    char name[20]; /**< The team's name */
    int noteWin; /**< Points won by this team  */
    int noteLost; /**< Points lost by this team */
}Team;
```

Elle contient le **nom** de « Team », la **note qu'elle a déjà gagnée** et la **note qu'elle a déjà perdue**.

# 2 Conception

## 2.1 Logique de la réalisation du projet

1. Tout d'abord, on va allouer une mémoire partagée pour stocker ce tableau de type 'Team', et on va aussi demander un mutex pour éviter plusieurs processus de manipuler un même fichier ou un même tableau, etc.

2. Ensuite, pour la règle de compétition, on vous montre une image comme Figure 1, ça c'est juste un exemple, mais c'est plus clair. Par exemple, on a 16 teams, de T0 à T15, et on suit le mécanisme d'élimination. Dans 1 tour, 16 équipes s'affronteront deux par deux, le vainqueur accédant au tour suivant et l'équipe perdante arrêtant la compétition. Ainsi de suite, jusqu'à le winner est produit. Dans notre cas, winner est T14.

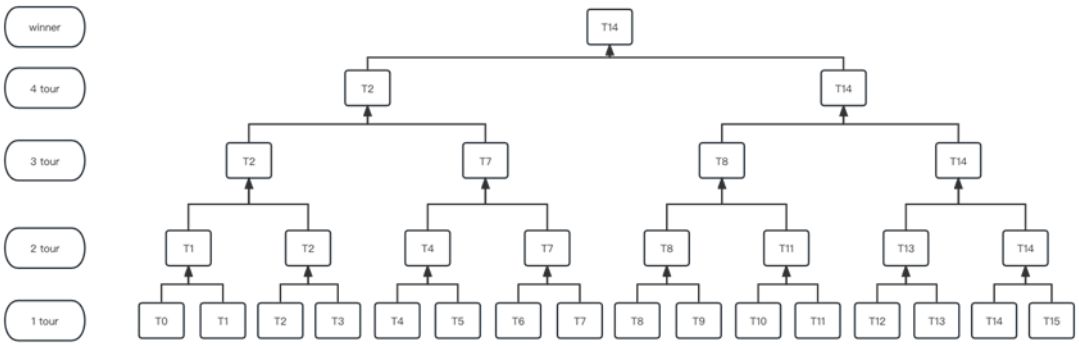


Figure 1 - Démonstration de compétition

3. Et puis, par la logique, chaque match est un processus, lorsque ce match termine, ce processus va attendre pour gagner le mutex pour écrire les informations détaillées dans un fichier 'result.txt'.

4. Comme nous suivons le système d'élimination et que le nombre d'équipes est une puissance de 2, le nombre de tours est prédéterminé et log2 (le nombre total d'équipes) est le nombre de tours joués comme Figure 1. C'est-à-dire après 4 tours, le winner va être certainement produit.

5. Comme il se doit, plusieurs tours de chaque tournoi peuvent être joués simultanément ; le tour N se termine et le tour N+1 commence immédiatement. Nous avons donc conçu le code du programme de manière à satisfaire logiquement le jeu simultané, mais avec un mécanisme légèrement différent. En effet, le fichier txt ne peut être ouvert pour modification que par un seul processus à la fois, sinon il y aurait des problèmes. Donc, nous utilisons le mécanisme 'mutex' pour s'assurer la sécurité de I/O.(Vous pouvez le voir dans les deux lignes '244' et '294', les opérations P et V). C'est pourquoi le terminal donne l'impression que le même tour est joué de manière séquentielle.

6. Dans la fonction 'match()', nous utilisons les 'printf()' avec la couleur pour illustrer le commencement de chaque match. Vous pouvez changer la couleur que vous préférez. Par exemple, dans la ligne 227 du fichier 'main.c' : printf("\033[43;31m\n\n%d tour\033[0m\n",tour); Vous pouvez changer comme ça : printf("\033[47;31m\n\n%d tour\033[0m\n",tour); Vous allez voir des chaîne avec fond blanc et police rouge.

7. Les deux fonctions 'findT1' et 'findT2', ça va être clair après nous vous montrons un exemple, par exemple, le 2ème tour commence maintenant, d'après Figure 1, nous voulons trouver 2 teams qui ont été gagné le match du 1er tour. Donc, 'findT1' trouve le team T1, et 'findT2' trouve le team T2 dans l'intervalle 0-3. C'est le même principe pour le 3ème tour, 'findT1' trouve le team T2 et 'findT2' trouve le team 'T7' dans l'intervalle 0-7.

# 3 Environement

## 3.1 Environement de développeur

Le projet a été écrit dans deux environnements de développement principaux. Pour le développeur, Yicheng Qian, l'environnement de développement utilisé est indiqué dans Figure 2 ci-dessous.

```
yc@qianyidengdeMacBook-Pro ~ % clang --version
Apple clang version 14.0.0 (clang-1400.0.29.202)
Target: arm64-apple-darwin21.6.0
Thread model: posix
InstalledDir: /Library/Developer/CommandLineTools/usr/bin
yc@qianyidengdeMacBook-Pro ~ %
```

Figure 2 - clang version

Pour le développeur, Linxiang CONG, l'environnement de développement utilisé est indiqué dans Figure 3 ci-dessous.

```
yoann@ubuntu:~$ gcc --version
gcc (Ubuntu 9.4.0-1ubuntu1~20.04.1) 9.4.0
Copyright (C) 2019 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

Figure 3 - gcc verison

# 4 Coopération

## 4.1 Répartition des tâches

Yicheng Qian est responsable de la mise en œuvre de l'espace partagé des processus, de la mise en œuvre du solitaire des processus et de la mise en œuvre des méthodes fonctionnelles. Il est également responsable de la rédaction des rapports.

Linxiang CONG est responsable de la conception du système de course et de la mise en œuvre de la boucle principale du programme, ainsi que de la rédaction des rapports et de la mise au point de la présentation du rapport.