

Systèmes informatiques

Projet 2 - Support pour l'interview de design

Justine Doutreloux
Nicolas Breels

27 avril 2015

1 Architecture

Il y a 6 fonctions de bases.

- **void** take_number(**int** number, **char** *file) : Cette fonction prend un nombre, elle le factorise en nombre premier. Ensuite elle écrit les différents nombres premier dans un fichier les uns à la suite de l'autres et elle termine en écrivant le nom du fichier. Si l'argument fichier est NULL, elle écrit : " noFile".
- **void** only_number(**char** *file) : Prend le fichier qui a été écrit par les différents threads, affiche le nombre unique à la première ligne, le nom du fichier à la deuxième et le temps d'exécution à la troisième.
- **void** take_fichier(**char** *file) : Prend un fichier, converti chaque nombre avec la librairie endian(3) et applique
take_number
- **int** max_threads() : retourne le nombre maximum de thread rentré par l'utilisateur.
- **void** add_number() : Applique
take_number
- **void** add_file() : Télécharge le fichier à l'adresse URL rentré par l'utilisateur et l'enregistre dans le dossier courant du projet. Applique
take_fichier

2 Threads

Le programme lance des threads qui lisent les fichiers et chargent les nombres dans une liste chaînée. Pendant le chargement des nombres, d'autres threads calculent les nombres premiers. Ensuite, on trouve le nombre qui est unique.

3 Synchronisation

Nous utilisons les sémaphores car nous les trouvons plus faciles à appliquer dans le cas qui nous occupe. Il faudra faire attention aux différentes utilisations de post et wait. Deux threads ne pourront pas écrire dans le fichier central en même temps.

4 Structures de données

Les facteurs premiers seront échangés via un fichier central dans lequel chaque thread écrira (synchronisation via les sémaphores).

5 Algorithme

```
1  #include <stdio.h>
2  unsigned int liste_facteurs[32];
3  int i = 0;
4  void f(unsigned int n, unsigned int d)
5  {
6      if (n != 1)
7      {
8          if (n % d)
9          {
10             d++;
11             f(n, d);
12         }
13         else
14         {
15             liste_facteurs[i] = d;
16             i++;
17             f(n / d, d);
18         }
19     }
20 }
21 int main(void)
22 {
23     unsigned int n;
24     int k;
25     scanf("%u", &n);
26     f(n, 2);
27     for (k = 0; k < i; k++)
28         printf("%d ", liste_facteurs[k]);
29     printf("\n");
30     return 0;
31 }
```

Source : <http://openclassrooms.com/forum/sujet/produit-de-facteur-premier-58167>.