Pràctica 3. Generació de nombres primers
Laboratori d'Aplicacions i Serveis Telemàtics

Josep Cotrina, Marcel Fernàndez, Jordi Forga, Juan Luis Gorricho, Francesc Oller

Introducció

Una forma d'obtenir nombres primers en paral·lel consisteix en disposar de múltiples processos treballadors que interaccionen amb un gestor central com es descriu a continuació. El gestor, de forma incremental, genera candidats a nombre primer pels processos treballadors que ho demanin; de igual forma guardarà els resultats tornats pels treballadors. Cada procés treballador realitza de forma cíclica els següents passos: demanar següent candidat, comprovar si és primer i tornar el resultat. Per cada sol·licitud de candidat, el gestor torna el següent nombre senar a partir de 5 en ordre creixent. El mateix gestor tornarà un 0 quan ja s'han obtingut els N primers nombres primers desitjats, és la manera de dir al treballador que no cal que continuí treballant. Per determinar si un candidat és primer, el treballador comprova si és divisible per tots els nombres primers més petits que l'arrel quadrada del candidat. Per fer aquest operació, cada treballador guarda una còpia parcial de la taula de nombres primers que està composant el gestor. Si el treballador necessita un nombre primer de la seva còpia de taula i aquest encara no hi és, el demana al gestor mitjançant l'índex de la taula. Si el gestor encara no disposa del primer demanat, aquest farà aturar al treballador a la espera del primer sol·licitat. Una vegada determinat si un candidat és primer o no, el treballador reporta el resultat al gestor. El gestor haurà d'atendre els resultats rebuts en el mateix ordre que va facilitar els candidats, amb la finalitat de escriure en ordre a la seva taula els nous primers obtinguts. Això vol dir que si un treballador reporta el resultat d'un candidat més gran que l'esperat, aquest treballador serà aturat i posteriorment despertat quan correspongui. Si el candidat esperat és primer s'introdueix a la taula i s'incrementa el corresponent índex, si no, es descarta el candidat i s'incrementa l'índex de següent esperat.

A continuació es proporcionen el fitxer Primers.java (principal) i els fitxers incomplerts Treballador.java i Gestor.java:

```
class Primers
{
  public static void main(String []args)
  {
    if (args.length!=2)
      {
        System.err.println("Arguments: num_primers num_treballadors");
        return;
      }

    int N = Integer.parseInt(args[0]);
    Gestor g = new Gestor(N);

    int M = Integer.parseInt(args[1]);
    Thread[] t = new Thread[M];
    for (int i = 0; i < M; i++)
      {
        t[i] = new Thread(new Treballador(N, g));
        t[i].start();
      }
    for (int i = 0; i < M; i++)
      {
```

```
      try {
        t[i].join();
      } catch(InterruptedException e) { }
    }
   for (int i = 0; i < N; i++)
    System.out.println("primers["+i+"] = "+g.primers[i]);
  }
}
```

```
class Treballador implements Runnable
{
 int[] primers;
 int ultim_index;
 Gestor gestor;

 Treballador(int N, Gestor g)
 {
  primers = new int[N];
  primers[0] = 3;
  ultim_index = 0;
  gestor = g;
 }

 public void run()
 {
  while (true)
    {
     int candidat, i;
     boolean es_primer;
     // Obté el seguent candidat (a completar)
    ...
    if (candidat==0)
       break; // acaba
     es_primer = true;
     i = 0;
     while (es_primer && (primers[i]*primers[i] <= candidat))
       {
        if (candidat % primers[i] == 0)
          es_primer = false; // candidat no es primer
        else
          {
           i++; // avanca al seguent index
           if (i > ultim_index)
             {
               int primer;
               // Obte del gestor el primer amb index i (a completar)
               ...
               ultim_index++;
               primers[ultim_index] = primer;
           }
        }
       }
    // Notifica al gestor el resultat del candidat processat (a completar)
    ...
```

```
    }
  }
}
```

```
class Gestor
{
  // camps de dades

  Gestor(int N /* nombre de nombres primers a calcular */)
  {
  }

  /* synchronized (sols Java natiu) */ int seguent_candidat()
  {
  }

  /* synchronized (sols Java natiu) */ int primer(int index)
  {
  }

  /* synchronized (sols Java natiu) */ void resultat(int candidat, boolean es_primer)
  {
  }
}
```

Es demana per aquesta pràctica:

1 Solució bàsica amb monitors

Completar la solució i provar-la obtenint els 100 primers nombres primers.

2 Treballadors no fiables

Suposeu que no ens refiem dels treballadors a l'hora de calcular si un candidat és primer o no. Modificar la solució per donar el mateix candidat a 3 treballadors i decidir si és primer o no per majoria.

3 Treballadors que no acaben

Suposeu que hi ha treballadors que no completen la seva feina, és a dir, treballadors que demanen un candidat a nombre primer i després mai arriben a tornar el resultat. Programar una solució fent servir un await() amb timeout que impedeixi que s'esperi indefinidament a rebre el resultat d'un candidat en concret.

4 Solució amb pas de missatges

Programeu la solució bàsica amb una arquitectura client/servidor multithreading aprofitant el monitor programat en el primer apartat.