Totty: a total order multicast

**Experiments. Set up the basic multicast system, and use the following test program to experiment with different values for Sleep and Jitter. Does it keep workers synchronized? Justify why. Note that we are using the name of the module (i.e. basic) as a parameter to the start procedure. We will easily be able to test different multicast implementations. Sleep stands for the average number of milliseconds the workers should wait until the next message is sent. Jitter stands for the average number of milliseconds of network delay.**

| SLEEP | JITTER | Sinconitcació |
| --- | --- | --- |
| 10 | 0 | Si |
| 50 | 0 | Si |
| 100 | 0 | Si |
| 250 | 0 | Si |
| 10 | 10 | No |
| 50 | 50 | No |
| 100 | 100 | No |
| 250 | 250 | No |
| 50 | 100 | No |
| 100 | 250 | No |

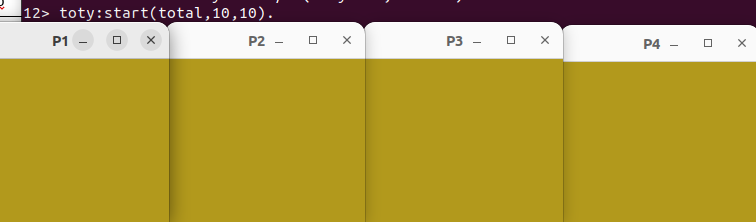
Variar el paràmetre SLEEP afecta la freqüència amb la qual els workers canvien el color. Com més gran sigui el paràmetre SLEEP més tardarà a canviar de color, i contra més petit més ràpidament canvien de color.  
 D’altra banda, canviar el paràmetre JITTER afecta a la sincronització, només estaran els workers sincronitzats al 100% quan aquest paràmetre sigui 0 i contra més gran sigui aquest valor més dessincronitzats estan.

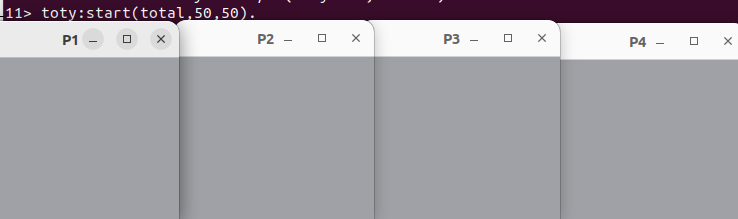
**i) Set up the total order multicast system, and repeat the previous tests. Does it keep workers synchronized?**

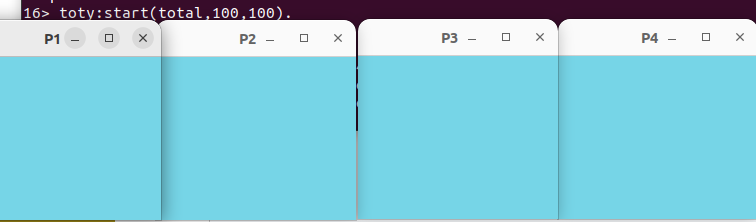
| SLEEP | JITTER | Sincronització |
| --- | --- | --- |
| 10 | 0 | Si |
| 50 | 0 | Si |
| 100 | 0 | Si |
| 250 | 0 | Si |
| 10 | 10 | Si |
| 50 | 50 | Si |
| 100 | 100 | Si |
| 250 | 250 | Si |
| 50 | 100 | Si |
| 100 | 250 | Si |

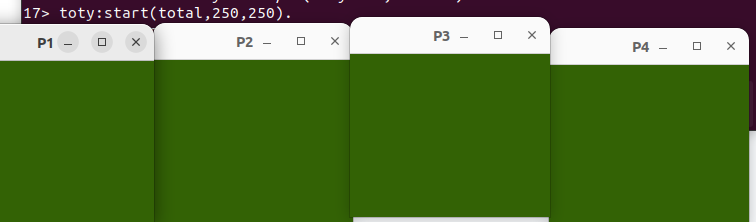
El sistema de Total Order Multicast garanteix la sincronització de tots els workers, independentment dels retards JITTER. Això s’aconsegueix mitjançant propostes de números de seqüència (proposals), la inserció ordenada a la cua, i acords finals (agreed).

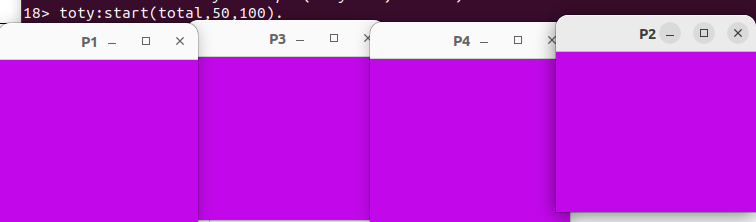
Aquest comportament assegura que els workers estiguin sincronitzats, a diferència del bàsic, que la presència de Jitter causava que els missatges arribessin en un ordre diferent a cada worker, generant dessincronització.

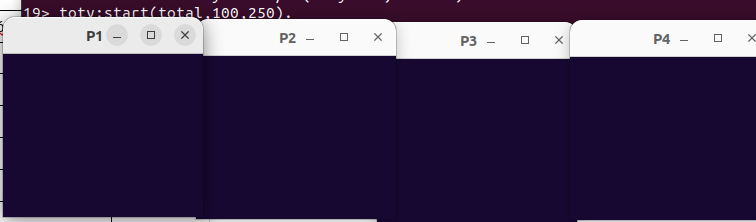
****

****

****

****

****

****

**ii) We have a lot of messages in the system. Derive a theoretical quantification of the number of messages needed to deliver a multicast message as a function of the number of workers and check experimentally that your formulation is correct.**

Formulació teorica:

A la fase de proposta, cada worker proposa un número de seqüència per al missatge multicast. Per tant, s'envien N missatges, un de cada worker.

A la fase d'acord, cada worker envia la seva proposta a tots els altres workers. Per tant, s'envien N × (N - 1) missatges. Doncs, tenim N+N\*(N-1) que això és igual a N²

Finalment, sumem un missatge final que indica que el missatge multicast s'ha entregat correctament, cosa que dona com a resultat total: N² +1

Per tant, el total de missatges necessaris per entregar un missatge de multidifusió és N² +1

Formulació experimental: hem retocat el codi per a que surtin els missatges que s’envien i hem obtingut aquest total



Primera execució:

* S'han enviat 255 missatges.
* Els workers han canviat de color 15 vegades.
* El càlcul és: 255 / 15 = 17, que coincideix amb la fórmula teòrica: 4² + 1 = 17.

Segona execució:

* S'han enviat 187 missatges.
* Els workers han canviat de color 11 vegades.
* El càlcul és: 187 / 11 = 17, que també coincideix amb la fórmula teòrica: 4² + 1 = 17.

Tercera execució:

* S'han enviat 136 missatges.
* Els workers han canviat de color 8 vegades.
* El càlcul és: 136 / 8 = 17, que torna a coincidir amb la fórmula teòrica: 4² + 1 = 17.

Els canvis de color s'han comptat manualment observant l'execució i anotant el nombre de vegades que els workers canvien de color.

**iii) Compare with the basic multicast implementation regarding the number of messages needed.**

A la implementació bàsica, quan un worker vol enviar un missatge a tots els altres workers, el retransmet a cada un d'ells. Doncs, per a cada missatge enviat, es realitzen N missatges, on N és el nombre de workers, en aquest cas 4.

A la implementació amb ordre total, els missatges s'han d’entregar en el mateix ordre a tots els workers, mitjançant un mecanisme de propostes i acords:

* Proposta: Cada worker proposa un número de seqüència per al missatge multicast. Això implica N missatges de proposta.
* Acord: Cada worker envia la seva proposta a tots els altres workers. Això comporta N × (N - 1) missatges.
* Missatge final: indica que el missatge multicast s'ha entregat. s’afegeix 1 missatge addicional.

Per tant, el total és: N+N\*(N+1)+1=N²+1

En la implementació bàsica, es necessiten menys missatges, però no es garanteix la sincronització, a no ser que el JITTER sigui 0. En canvi, la implementació amb ordre total necessita N² + 1 missatges, però garanteix al 100% la sincronització, tot i que a costa d’un major nombre de missatges.