Compléxité: Espace

The goal is to share the same vector bewtween all processus, so the choice of algorithm won't change the results. Let n be the number of processus and x the space taken by the vector in the system's memory:

O(n*x)

Compléxité: Temps

Here we'll witness different complexities for each algorithms.

Simple: Here we have a simple code that uses the command for that iterates the same code for all processus, we have then (n-1) cycles. Let each cycle be of time t, and n processus:

O((n-1)*t) , or -> O(n*t)

Anneau: It's here we see the interest of parallélisme, we are sending the same amount of information, but sinse we do it in parallel we do lesse cycles. Let time per cycle be t, and n the number of processus:

O(2*t + (n-3)/2 * t), or -> O((n/2)*t)

Hypercube: For this last one, the train of thought is the same, just that we have even more processus sending information:

cycle 1 -> we have 1 processus

cycle 2 -> we have 1 processus

cycle 3 -> we have 1 processus

...

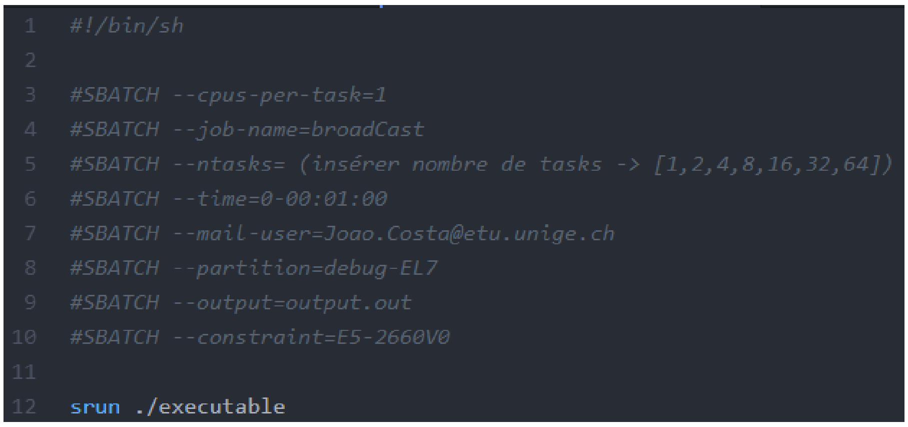
cycle n -> we have $2^{(n-1)}$ processus

For n processes we will require $log_2(n)$ cycles. time t per cycle

O($log_2(n)$*t)

Résultats

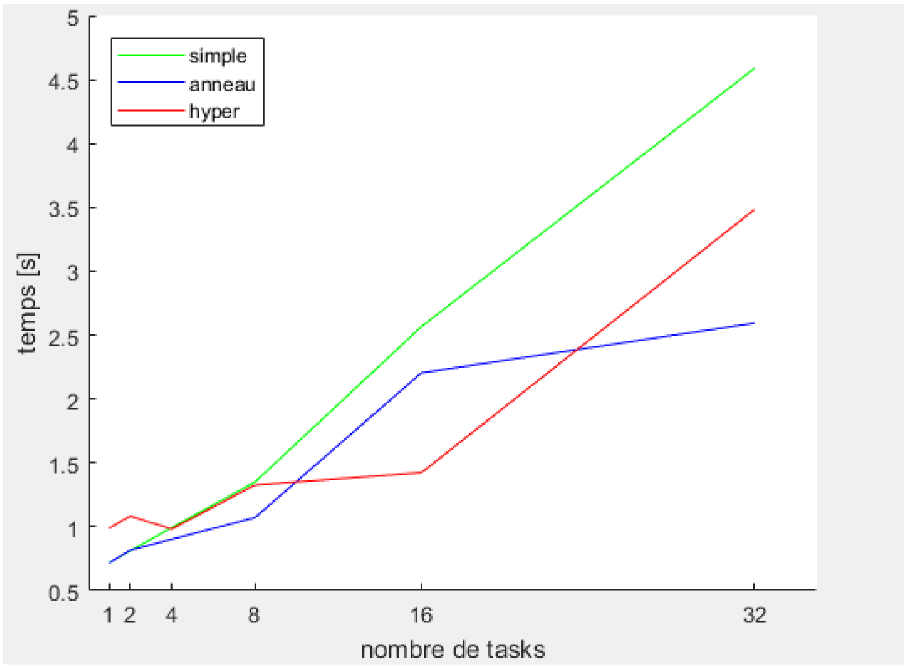
For the results I used the following code in the batchfile:



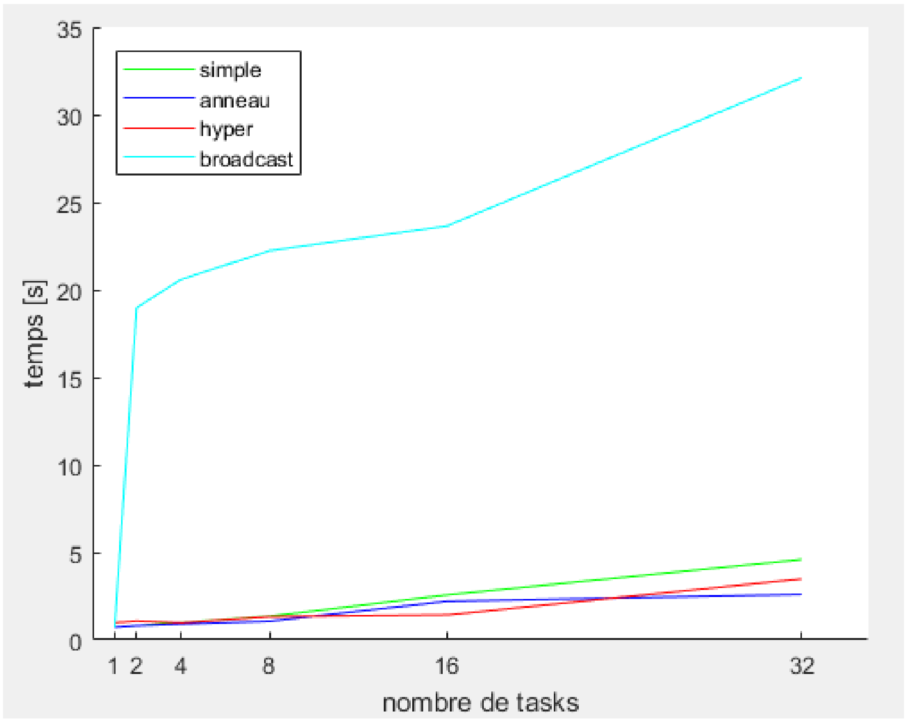
```
#!/bin/sh

#SBATCH --cpus-per-task=1
#SBATCH --job-name=broadCast
#SBATCH --ntasks= (insérer nombre de tasks -> [1,2,4,8,16,32,64])
#SBATCH --time=0-00:01:00
#SBATCH --mail-user=Joao.Costa@etu.unige.ch
#SBATCH --partition=debug-EL7
#SBATCH --output=output.out
#SBATCH --constraint=E5-2660V0

srun ./executable
```

I had problems to execute this batchfile with nProc = 64, I discussed them with you via email.

For simple, Anneau and Hypercube, I find unexpected results, as Anneau became faster than Hyper as n grew larger. Simple is as expected.



I don't know why but using the simple BCast command didn't work as intended, it was very slow, I don't know if it was just by chance, but I did try it many times, and I always got the same results



I don't want to talk much about the broadcast function results, because they are probably not accurate. However, I theorised that Hypercube algorithm would be less expensive than Anneau, which it is at the beginning, but when we cross the 32 cores mark, Anneau becomes faster, this was theoretically unexpected, but we were told it would be the result.
My best guess is that we are not taking full advantage of the structure/architecture, and that we will later on find an algorithm that exploits the same idea but does it in another way, because intuitively Hypercube looks much faster than Anneau.