

Report : Parallel computation of π decimals

Alexandre Temperville, Master 2 Scientific Computing

All that we need to deal with this report is in the folder 'Pi'.

The first π decimals

The Leibniz formula gives π :

$$\pi = 4 \arctan(1) = 4 \sum_{k=0}^{+\infty} \frac{(-1)^k}{2k+1}.$$

Let us define f such that $f(k) = 4 \frac{(-1)^k}{2k+1}$.

As $|f(k)| > |f(k+1)| \quad \forall k \in \mathbb{N}$, and as this sum is obviously an alternating one, we can claim that it converges towards a real number : π . This is a well-known result known as the Leibniz test (or in French: "théorème de convergence des séries alternées").

Let us denote

$$\begin{aligned} S_n &= 4 \sum_{k=0}^n \frac{(-1)^k}{2k+1} = \sum_{k=0}^n f(k) \quad \text{and} \\ R_n &= 4 \sum_{k=n+1}^{+\infty} \frac{(-1)^k}{2k+1} = \sum_{k=n+1}^{+\infty} f(k) \end{aligned}$$

We have :

$$\forall n \in \mathbb{N}, \quad \pi = S_n + R_n.$$

The Leibniz test also states that :

$$\forall n \in \mathbb{N}, \quad |R_n| < |f(n+1)| \quad \text{i.e. } R_n \text{ is bounded by } |f(n+1)|.$$

We can take n such that $|f(n+1)| < 10^{-i}$, then :

$$|R_n| = |\pi - S_n| < 10^{-i}, \quad \text{i.e. } S_n \in [\pi - 10^{-i}, \pi + 10^{-i}].$$

Under this condition S_n gives an approximation of π with $i-1$ exact decimals. We will use this condition to stop the computation of the sum with a precision of i decimals :

$$\begin{aligned} (|f(n+1)| < 10^{-i}) &\iff \left(\frac{4}{2n+3} < 10^{-i} \right) \iff \left(\frac{4 \times 10^i - 3}{2} < n \right) \iff \left(2 \times 10^i - \frac{3}{2} < n \right) \\ &\iff (2 \times 10^i - 1 \leq n) \end{aligned}$$

If we compute S_n with n such that the previous inequality is verified, then we will have an exact approximation of π up to $i-1$ decimals.

Algorithm in parallel

This algorithm is also explained in french in the program *pi.c*.

To get an approximation of π up to $i - 1$ decimals, where i is given as a parameter by the user when running the program, we will share out the calculation of the sum between processors, with the number of processors given as a parameter too.

We can choose $n = 2 \times 10^i - 1$ to satisfy the condition of the previous part. We can see that we need 2×10^i terms (variable *nb_termes*) of the sum to reach such precision. n chosen like this allows to find easily a regular split of the sum to do in each processor. As we cannot parallelize the sum asking different processors to compute one decimal for example, we need to separate computations to get each decimals when we have the total sum of the terms.

To get $i - 1$ exact decimals : if the number of processes is P , and if the rank of a process is *rank*, each process will compute a sum of $\frac{2 \times 10^i}{P}$ terms and stores the result in a variable *partial_sum*.

Finally, we do a reduction with add in process of rank 0, that will compute the result of π with the precision we want thanks to the variable *total_sum* and return it on the screen and in a file *pi.dat*.

Execution of the program

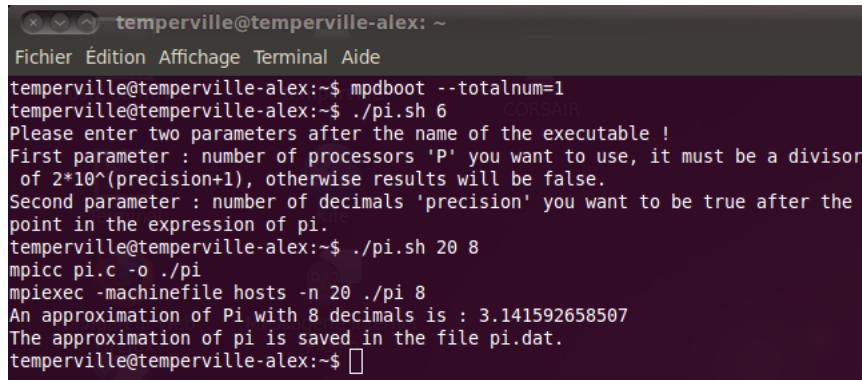
I tried to test my program on **g5k** in the frontend of Lille, and we obtain the results described in the picture *g5k.png*. We see with the *ls* command that the results are saved in the file *pi.txt*, I have some warning for *malloc*, I do not understand why.

I follow the instructions in the following web-site to work with MPI successfully :

http://www.grid5000.fr/mediawiki/index.php/Run_MPI_On_Grid'5000

To do that, on the frontend of Lille, I make a reservation : *oarsub -I -t allow_classic_ssh -l nodes=3*. Then, I enter the instructions we can see in the picture *g5k.png*.

Here we can see what I obtain in my own computer :



```
temperville@temperville-alex: ~  
Fichier Édition Affichage Terminal Aide  
temperville@temperville-alex:~$ mpdboot --totalnum=1  
temperville@temperville-alex:~$ ./pi.sh 6  
Please enter two parameters after the name of the executable !  
First parameter : number of processors 'P' you want to use, it must be a divisor  
of 2*10^(precision+1), otherwise results will be false.  
Second parameter : number of decimals 'precision' you want to be true after the  
point in the expression of pi.  
temperville@temperville-alex:~$ ./pi.sh 20 8  
mpicc pi.c -o ./pi  
mpiexec -machinefile hosts -n 20 ./pi 8  
An approximation of Pi with 8 decimals is : 3.141592658507  
The approximation of pi is saved in the file pi.dat.  
temperville@temperville-alex:~$
```

On my computer, I run the script *pi.sh* which compiles and the file *pi.c* and runs the executable *pi* with 2 parameters, the first one is the number of processors to use and the second one is the number of decimals we want to be exact. I notice that when I want more than 8 exacts decimals, it is too for my computer (too many computations and precision too high).