With TWO threads…

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

The dividers utilization is high but not so high, because they interfere with each other.

With just one thread indeed we can observe that the divider utilization is very high! (and the execution time is approximately the same)

Immagine che contiene testo, schermata, Parallelo, design

Descrizione generata automaticamente

From here: <https://stackoverflow.com/a/48334134>

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

And using

double float\_mod = (double) fmod(num, i);

if (float\_mod == 0.0) {

cout << num << " and i: " << i << endl;

}

Instead of the classical % we can see from the assembly code that the floating point division is used…

Immagine che contiene testo, schermata, numero, software

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Parallelo, software

Descrizione generata automaticamente

Indeed, in the case of both fdiv and div, the integer dividers have a utilization of just 28.2%.

Now we try to introduce a dummy wait after the integer dividers to see what happens.

The introduced code is the following:

// Dummy wait (sleep)

bool condition = (num % i) == 0;

// this\_thread::sleep\_for(chrono::nanoseconds(1));

int j, sum = 0;

for (j = 0; j < 99; j++)

sum += j;

if(condition) cout << "J: " << j << " sum: " << sum<<endl;

if (condition) {

cout << "I: " << i << endl;

…

}

AND as we were thinking, now the utilization of the dividers is very low!

Immagine che contiene testo, schermata, Parallelo, diagramma

Descrizione generata automaticamente

What we must do is the plot.. To see if 1 and 2 are equals… like before or not

Immagine che contiene testo, Diagramma, linea, numero

Descrizione generata automaticamente

And the cumulative speed-up:

Immagine che contiene linea, testo, Diagramma, diagramma

Descrizione generata automaticamente

Base code used: new\_trial\_division.cpp

Domande prof:

Introdurre una attesa dopo dell'operazione di modulo per diminuire l'utilizzazione della DIV.

Fornire un controesempio che vada a giustificare le nostre osservazioni.

Provare a fare divisione su interi anche.

Vedere la mod come viene implementata a basso livello. Se fa uso di divisori in Floating Point o di divisori tra interi.

Provare anche fmod

Vedere se in assembly c'è una DIV o FDIV

This instead is with one thread per core alone, when there are less than 6 threads. (pool\_affinity\_core in the csv)

Immagine che contiene linea, Diagramma, diagramma, testo

Descrizione generata automaticamente

And here the Divider utilization is REALLY high, in the order of 92%.

(Eventualmente) provare ad utilizzare il co-processore SIMD disponibile all’interno della CPU per eseguire in parallelo queste operazioni complesse. Oppure provare ad implementare la mod come ciclo while. 5%2: 5-2=3-2=1!=0.

…