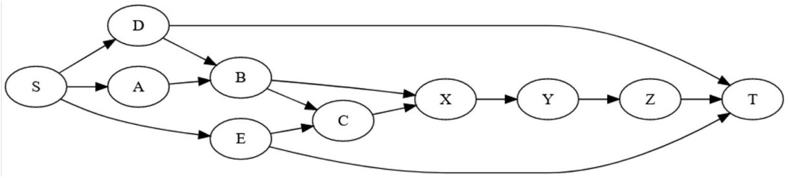
PARALLEL PROGRAMMING...

By Patrick Lemoine 2024.

Class Taskflow_HPC

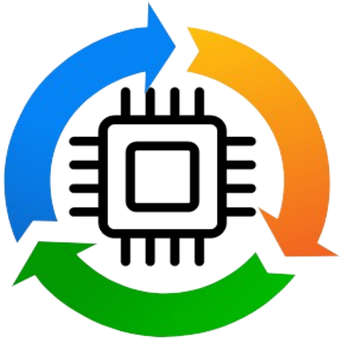
```
myFunctionLambda 1=[...](const int &k) {…}
myFunctionLambda 2=[...](const int &k) {...}

Taskflow_HPC myTask( NbThread, NumType );

//NumType 0: NoThread, 1: multithread,  2: std::async, 3: Specx, 4: jthread...

myTask.add(
    _parameters=Frontend::parameters(valnput1,...,valOutput1,...),
        _task=myFunctionLambda 1);

myTask.add(
    _parameters=Frontend::parameters(valnput1,...,valOutput1,..),
        _task=myFunctionLambda 2);

myTask.run();

myTask.close();

myTask.debriefing();
```

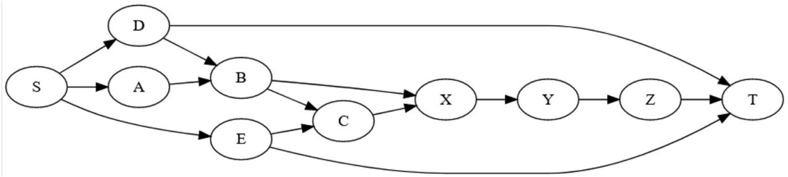


Detach

```
myFunctionLambda 1=[...](const int &k) {…}
myFunctionLambda 2=[...](const int &k) {…}
myFunctionLambda 3=[...](const int &k) {...}

Taskflow_HPC myTask( NbThread, NumType );

//NumType 0: NoThread, 1: multithread,  2: std::async, 3: Specx, 4: jthread...

myTask.add(
    _parameters=Frontend::parameters(valnput1,...,valOutput1,...),
        _task=myFunctionLambda 1);

myTask.qDetach=true;

myTask.add(
    _parameters=Frontend::parameters(valnput1,...,valOutput1,...),
        _task=myFunctionLambda 2);

myTask.add(
    _parameters=Frontend::parameters(valnput1,...,valOutput1,..),
        _task=myFunctionLambda 3);

myTask.run();

myTask.close();

myTask.debriefing();
```

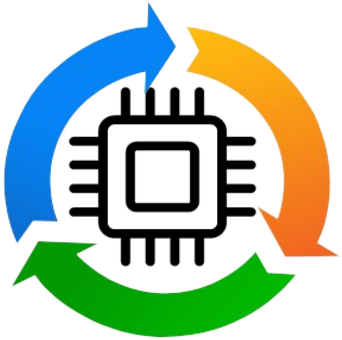
Detach Part

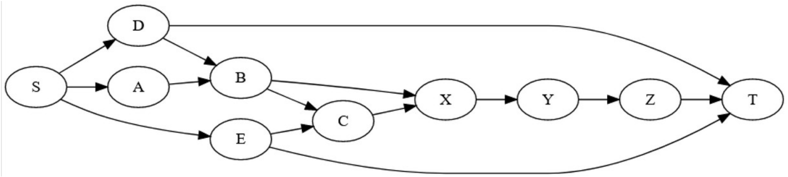




For_each

```
myFunctionLambda 1=[...](const int &k) {…}
myFunctionLambda 2=[...](const int &k) {…}

Taskflow_HPC myTask( NbThread, NumType );

//NumType 0: NoThread, 1: multithread,  2: std::async, 3: Specx, 4: jthread...

std::vector<int> v(nb,0);

myTask.for_each(v.begin(),v.end()
    _parameters=Frontend::parameters(valnput1,...,valOutput1,...),
        _task=myFunctionLambda 1);

myTask.run();

myTask.for_each(v.begin(),v.end()
    _parameters=Frontend::parameters(valnput1,...,valOutput1,...),
        _task=myFunctionLambda 2);

myTask.run();

myTask.close();

myTask.debriefing();
```





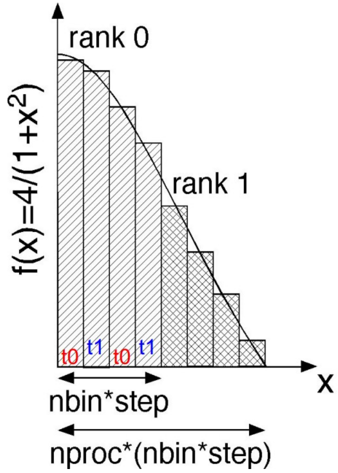
Compute π with Taskflow_HPC

GOAL : The following code computes the π number by using a numerical evaluation of an integral by a rectangle method.

$$\pi = \int_0^1 \frac{4}{1+x^2}\,dx \cong \Delta \sum_{i=0}^{N-1} \frac{4}{1+x_i^2}$$

```
void MyDemo()
{
   std::cout<<"\n";
   int nbThreads=9;
   long int nbN=1000000;
   int sizeBlock=nbN/nbThreads;
   double h=1.0/double(nbN);

   double integralValue=0.0;
   std::vector<double> valuesVec(nbThreads,0.0);


   auto FC1=[h,sizeBlock](const int k,double& s) {
         int vkBegin=k*sizeBlock;
         int vkEnd=(k+1)*sizeBlock;
         double sum=0.0; double x=0.0;
         for(int j=vkBegin;j<vkEnd;j++) { x=h*double(j); sum+=4.0/(1.0+x*x); }
         s=sum;
      return true;
   };
```

```
Taskflow_HPC Test1(nbThreads,3);

Test1.qSave=true;

Test1.setFileName("./DATA/PI");


for(int k=0;k<nbThreads;k++) {
            auto const& idk = k;
            Test1.add(_parameters=Frontend::parameters(idk,valuesVec.at(idk)),_task=FC1);
}

...


   Test1.run();
   Test1.close();

    std::cout<<"\n";
    integralValue=h*std::reduce(valuesVec.begin(),valuesVec.end());
    std::cout<<"PI Value= "<<integralValue<<"\n";

    std::cout<<"\n";
    Test1.debriefingTasks();
}
```



Thank you for your attention !