Calcul d’intégrale

par la méthode des trapèzes

Une image contenant texte, carte

Description générée avec un niveau de confiance très élevé

Test préalable généré depuis un programme python personnel

Table des matières

[Introduction et contexte 3](#_Toc503197682)

[Présentation du sujet 3](#_Toc503197683)

[Description de la machine de test 3](#_Toc503197684)

[Documentations 3](#_Toc503197685)

[Algorithme 4](#_Toc503197686)

[Principe et fonctionnement séquentiel 4](#_Toc503197687)

[Parallélisation sans OpenMP 4](#_Toc503197688)

[Parallélisation avec OpenMP 4](#_Toc503197689)

[Comportement et évolution du programme 5](#_Toc503197690)

[Évolution séquentielle 5](#_Toc503197691)

[Évolution parallèle sans OpenMP 5](#_Toc503197692)

[Évolution parallèle avec OpenMP 5](#_Toc503197693)

[Conclusion 6](#_Toc503197694)

[Avantages et inconvénients de la méthode 6](#_Toc503197695)

[Commentaires sur le codage 6](#_Toc503197696)

# 

# Introduction et contexte

# 

## Présentation du sujet

Le but du projet est de calculer l’intégrale d’un polynôme entre 2 bornes par la méthode des trapèzes, pour un nombre de threads et d’itérations donnés.

La formule de la courbe est stockée dans une structure par le biais d’un simple tableau d’entier.

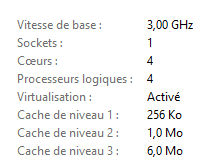
Ex : coef[0] = a ; coef[1] = b ; coef[2] = c ;

Le programme résulte par l’affichage des résultats en console, ainsi que la création d’un fichier txt contenant ces mêmes résultats.

## Description de la machine de test

Pour les besoins des tests, le programme ‘plot\_curve.cpp’ sera utilisé car il illustre bien l’utilisation du processeur pendant les calculs, puisqu’il lance de manière successive le programme de calcul principal avec différents paramètres.

* Description de la machine : PC fixe (personnel)



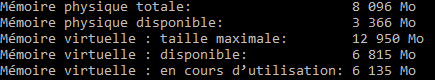
* Évolution du processeur en temps réel pendant le programme (plotCurveExec) :



(double clic pour ouvrir la vidéo)

* RAM : Obtenue avec la commande « systeminfo » sous Windows

Avant le lancement du programme



Pendant le programme

Une image contenant intérieur

Description générée avec un niveau de confiance élevé

## Documentations

* **Calcul d’intégrales par la méthode des trapèzes :**

Approximation d’une intégrale, « Algorithmique et calcul numérique », José OUIN

<https://fr.wikipedia.org/wiki/M%C3%A9thode_des_trap%C3%A8zes>

<http://serge.mehl.free.fr/anx/meth_trap.html>

<http://www.courspython.com/integration-rectangle-trapeze.html>

* **Parallélisation du programme :**

Cours sur Amétice

https://www.lri.fr/~falcou/teaching/par/OpenMP-cours.pdf

# Algorithme

## Principe et fonctionnement séquentiel

## Parallélisation sans OpenMP

## Parallélisation avec OpenMP

# 

# Comportement et évolution du programme

Les courbes et les explications ci-après concernent les programmes integrale\_sequentiel.cpp / integrale\_thread.cpp / Projet\_OpenMP.cpp

## Évolution séquentielle

D’après la courbe ci-dessus, il semble que le programme séquentiel perd en efficacité lorsque le nombre d’itérations tend à augmenter.

En effet, on se trouve plutôt en O(n²) au niveau du temps d’exécution.

## Évolution parallèle sans OpenMP

Cette fois-ci, la courbe indique une certaine stabilité au niveau des temps d’exécution (même si certains sont plus ou moins étranges).

Il est possible de dire une telle chose si l’on prend en compte les unités (ordre de grandeur), et non de l’allure de la courbe.

## Évolution parallèle avec OpenMP

Pour la version OpenMP, les courbes sont quasiment confondues.

Il est remarquable, bien que l’allure de la courbe montre une hausse du temps d’exécution, que les unités ont un ordre de grandeur bas et offrent malgré tout plus de performance que les autres programmes.

# Conclusion

## Avantages et inconvénients de la méthode

## Commentaires sur le codage