Práctica 3
Paralelismo a nivel de hilos
Parte I: introducción a OpenMP

Objetivos:

- Aprender a paralelizar una aplicación en una máquina paralela de memoria centralizada mediante hilos (*threads*) usando el estilo de *variables compartidas*.
- Estudiar el API de OpenMP y aplicar distintas estrategias de paralelismo en su aplicación.
- Aplicar métodos y técnicas propios de esta asignatura para estimar las ganancias máximas y la eficiencia del proceso de paralelización.
- Aplicar todo lo anterior a un problema de complejidad y envergadura suficiente.

Desarrollo:

En esta práctica los/las estudiantes deben paralelizar usando OpenMP la solución a un problema dado para aprovechar los distintos núcleos de los que dispone cada ordenador de prácticas. Se paralelizará por tanto para un sistema multiprocesador (máquina paralela de memoria centralizada), en el que todos los núcleos de un mismo encapsulado ven la misma memoria, es decir, un puntero en un núcleo es el mismo puntero para el resto de los núcleos del microprocesador.

La práctica está dividida en 2 partes. La primera, esta, está pensada para que los alumnos se familiaricen con OpenMP y las posibilidades que ofrece para paralelizar soluciones a problemas que se puedan utilizar en sistemas multiprocesador. En la segunda parte se propondrá un problema concreto para cuya solución y estudio se utilizará lo aprendido en la primera parte.

Tarea 0 Entrenamiento previo:

Observe el siguiente programa en C donde se suman dos vectores de *floats* empleando OpenMP para paralelizar el cálculo.

```
#include <omp.h>
#define N 1000
#define CHUNKSIZE 100

main(int argc, char *argv[]) {

        int i, chunk;
        float a[N], b[N], c[N];

        /* Inicializamos los vectores */
        for (i=0; i < N; i++)
          a[i] = b[i] = i * 1.0;
        chunk = CHUNKSIZE;

        #pragma omp parallel shared(a,b,c,chunk) private(i)
          {
                  #pragma omp for schedule(dynamic,chunk) nowait
                  for (i=0; i < N; i++)
                    c[i] = a[i] + b[i];

          }   /* end of parallel region */

}
```

Revise la documentación de OpenMP (API, tutoriales, como este del Lawrence Livermore Computing Center, etc...) y responda:

0.1 ¿Para qué sirve la variable **`chunk`**?

0.2 Explique **completamente** el *pragma* :

```
#pragma omp parallel shared(a,b,c,chunk) private(i)
```

- ¿Por qué y para qué se usa **`shared(a,b,c,chunk)`** en este programa?
- ¿Por qué la variable `i` está etiquetada como **`private`** en el pragma?

0.3 ¿Para qué sirve **`schedule`**? ¿Qué otras posibilidades hay?

0.4 ¿Qué tiempos y otras medidas de rendimiento podemos medir en secciones de código paralelizadas con OpenMP?

Tarea 1: Estudio del API OpenMP [Parte individual obligatoria (25% de la nota)]

Se deberá estudiar el API de OpenMP y su uso con GNU GCC, comprobando el correcto funcionamiento de algunos de los ejemplos que hay disponibles en Internet (p.ej.página sobre programación paralela de la Universidad de Granada)

Cada miembro del grupo* deberá realizar un pequeño tutorial a modo de "libro de recetas de paralelismo con OpenMP" con **distintos ejemplos de aplicación** del paralelismo que ofrece OpenMP en función de distintas **estructuras de software**.

Notas generales a la práctica. Parte 1:

- Las respuestas y la documentación generadas tanto en las tareas de la parte I como de la II se integrarán en la memoria conjunta que se entregará al final de la práctica 3.
- Para el trabajo en la parte II y en la memoria se utilizará la plataforma de desarrollo GitHub. Esta permitirá una gestión de un repositorio común de trabajo que deberá crear cada grupo incluyendo a sus miembros y al profesor. Este dará a conocer su usuario en dicha plataforma para ser añadido.
- La implementación realizada tendrá que poder ejecutarse bajo el sistema operativo Linux del laboratorio de prácticas, aunque en casa puede trabajar con otros compiladores y sistemas operativos que soporten OpenMP (icc, clang, Visual C++, ...)
- Las/los estudiantes entregarán, además de las aplicaciones desarrolladas, una memoria, estructurada según indicaciones del profesor, con la información obtenida en las partes I y II, incluyendo todos los apartados individuales como anexo.
- La información referente a OpenMP se encuentra en www.openmp.org. Para compilar use gcc o g++ con el modificador `-fopenmp`.
- La práctica se deberá entregar mediante el método que escoja su profesor de prácticas antes de la sesión de prácticas de la semana del 22 de noviembre de 2021.

***Nota:**

Los trabajos teóricos/prácticos realizados han de ser originales. La detección de copia o plagio supondrá la calificación de "0" en la prueba correspondiente. Se informará la dirección de Departamento y de la EPS sobre esta incidencia. La reiteración en la conducta en esta u otra asignatura conllevará la notificación al vicerrectorado correspondiente de las faltas cometidas para que estudien el caso y sancionen según la legislación (Reglamento de disciplina académica de los Centros oficiales de Enseñanza Superior y de Enseñanza Técnica dependientes del Ministerio de Educación Nacional BOE 12/10/1954).