Universidad de Costa Rica

Escuela de Ciencias de la Computación e Informática

Examen 2

Respuestas Teóricas

---

jaccard_omp

- Explique cada una de las directivas de OpenMP que utilizó y cómo funcionan dentro del programa

```
#pragma omp parallel for
```

Lo anterior especifica que es un ciclo for y que debe dividir sus iteraciones en hilos para que corran en paralelo.

```
default(none) private(i) shared(matrix_row,test_vector)
```

El uso de default(none) obliga a especificar las variables que serán utilizadas como memoria compartida, en este caso serán matrix_row que tiene que ser compartida para que cada hilo en el for pueda comparar si ya llegó a su última iteración o no. Además debe usar esa variable para hacer las operaciones lógicas en los ifs del ciclo. El i es privado porque cada hilo trabaja con su propia iteración i y no deben tener el mismo varos hilos.

```
reduction(+: words_union, words_intersection) \
        schedule(static,1)
```

Se utiliza un reduction para que se divida entre los hilos y el schedule es estático y de forma que le toque una iteración a cada hilo. Se usa una reducción en

words_union y words_intersection ya que son los datos que están editando en el ciclo con una suma.

- Explique de forma general cómo se está dando la repartición de tareas entre hilos en su solución

Al usar el schedule de forma estática, la división de tareas es de forma que se asigna antes de la ejecución del programa para que a cada hilo le toque una iteración.

---

jaccard_mpi

- ¿Cuál o cuáles funciones de MPI consideró necesarias para la solución de este problema y explique cómo funciona(n) en su código?

```
MPI_Allreduce(&max_jaccard_similarity_local, &max_jaccard_similarity_global, 1, MPI_DOUBLE,
MPI_MAX, MPI_COMM_WORLD);
```

Se utilizó únicamente la función ALLREDUCE dentro del código paralelo, de forma que cada proceso calcula una similaridad de jaccard local, y la guarda en una variable la cuál envía para la reducción y el resultado se guarda en max_jaccard_similarity_global, el cual se imprime al final del programa.

- Explique cómo realizó la repartición de tareas entre los procesos.

Las tareas se reparten de forma que se tiene un contador que lleva el control de los procesos. Los procesos se rotan de forma circular, usando su identificador “myid” . Si tengo N procesos, a cada proceso le toca una línea, y voy incrementando el contador. Cuando dicho contador llega a N, lo devuelvo a 0, deja que los procesos corran de manera concurrente del 0 a N. Cuando llega a N, repite el mismo proceso, mientras hayan líneas.