Modelo de planificación y asignación de tareas a una máquina de ocho núcleos buscando un tiempo mínimo de ejecución

Beatriz Alejandra García[1]

Facultad de Ingeniería Mecánica y Eléctrica (FIME)
Universidad Autónoma de Nuevo León (UANL)
Av. Universidad, Cd. Universitaria, San Nicolás de los Garza, NL, México
beatriz.ale.gr@hotmail.com

Resumen

Este artículo presenta los resultados obtenidos a partir de una simulación en código `R` de la asignación de tareas que se requieren ejecutar en una máquina que tiene ocho núcleos de capacidad. La asignación es realizada conforme se va terminando alguna de las tareas y queda disponible ese núcleo.

Palabras clave: asignación de tareas, tiempo mínimo, simulación.

---

1. Introducción

El estudio de este problema es sobre la distribución de tareas que se deben realizar en una máquina que tiene como capacidad ocho núcleos, el motivo del estudio es para comprender de una mejor manera cómo es que trabaja un sistema de cómputo cuando se realiza una paralelización en el planteamiento de un código de una simulación de algún problema de estudio donde se puede asignar alguna tarea a alguno de los procesadores de ese sistema de cómputo ya que estos se activan al mismo tiempo y las tareas también se van realizando al mismo tiempo en cada uno de ellos; algunas de las tareas necesitan de uno o más procesadores para su realización por lo que el tema de estudio en este artículo es sobre cuáles tareas requieren de ciertos núcleos, en cuánto tiempo se ejecutan y cuál fue el tiempo máximo de ejecución al finalizar con la realización de las tareas.

La idea que se busca experimentar es el asignar tareas en el primer paso de la ejecución de manera que todos los procesadores trabajen con una tarea y conforme vayan

terminando alguna de ellas buscar otra tarea que pueda ser ingresada al procesador disponible y continuar con esta idea hasta que todas las tareas hayan sido realizadas.

El objetivo principal de la experimentación es encontrar el tiempo mínimo de ejecución, por lo cual se requiere conocer el procesador que tardó más en realizar las tareas que se le asignaron.

Las siguientes secciones muestran los antecedentes que se necesitan conocer para comprender de una mejor manera lo que se trabaja en el artículo, la solución propuesta para poder llevar a cabo la experimentación y también la evaluación de la experimentación con respecto a los resultados obtenidos y detalles que surgieron en el proceso de búsqueda de soluciones.

2. Antecedentes

Un problema de asignación de tareas tiene como objetivo buscar la manera de emparejar ciertas tareas a personas o máquinas que estén disponibles para realizarlas buscando minimizar costos, tiempos de espera o el tiempo total de ejecución. Para ello se supone que cada recurso se debe destinar solamente a una tarea y de la misma manera cada tarea solo puede ser ejecutada por un recurso.

Durante la revolución industrial (1760 – 1840), al ser un tiempo en que surgieron las máquinas, se tuvo la necesidad de asignar una tarea a un trabajador y se tenían ciertos métodos para realizarlo; sin embargo, formalmente apareció el problema hasta el año 1941 gracias a que F.L. Hitchcook, conocido por su formulación del problema de transporte, publicó una solución analítica del problema. Más tarde Harold W. Kuhn, matemático estadounidense que estudió la teoría de juegos, planteó el método húngaro en 1955 y éste fue revisado por James Munkres en 1957, quien es conocido por sus grandes conocimientos en el área de topología.

Hoy en día el problema de asignación es utilizado con frecuencia en la solución de problemas en la industria y es parte fundamental de la rama de investigación de operaciones dado que tiene como objetivo ayudar en la toma de decisiones en la organización de un sistema.

Sin dejar a un lado el tema a tratar, los problemas de asignación de tareas a procesadores constituyen – de acuerdo a la definición de Ezzatti [1] – "una subclase de los problemas de asignación de recursos con restricciones".

3. Metodología utilizada

La herramienta utilizada para la solución del problema a tratar es el programa `R`, el cual permite de una manera accesible realizar la codificación del modelo deseado y obtener resultados acertados, además es una herramienta completa ya que permite realizar gráficos de forma fácil y permite trabajar con tablas donde se pueden guardar y obtener datos requeridos para la solución del problema. La manera en la que se trabajó en la codificación del problema fue el obtener las características que deben tener las tareas que se desean asignar, es decir, los núcleos necesarios para ser procesado y el tiempo que lleva ejecutar esa tarea. Por otro lado se obtiene la capacidad máxima que

cada procesador tiene para llevar a cabo la ejecución total, la cual tiene una copia ya que al comparar los datos originales con los datos al término de la ejecución se obtienen las diferencias que son el tiempo total de ejecución de cada procesador. Una vez generados los datos que son requeridos para el procesamiento del problema de asignación se tiene un intervalo de pasos necesarios, los cuales marcarían una unidad de tiempo haciendo la simulación de que el tiempo fuera tomado en segundos, que permiten realizar las distintas asignaciones. Dado el paso uno se tiene que todos los núcleos deben recibir una tarea de cierto tiempo de ejecución requerido, una vez que se asignan tareas a todos los núcleos la información se guarda en una tabla que recibe como datos el número de tarea asignada, el núcleo en que se ingresó, el tiempo de ejecución total de la tarea, el paso en que entró al núcleo, el siguiente dato es el tiempo que lleva dentro del núcleo (que va a ser un contador que va a aumentar en uno cada vez que la tarea siga ejecutándose cada vez que se realiza un paso), una vez que haya salido la tarea se asigna el paso en que sale en la siguiente columna y por último se tiene si el núcleo está disponible o no, después de que todo se ha ingresado a la tabla donde se requieren estos datos se sigue con el paso dos, el cual verifica si existe o no un núcleo disponible (dependiendo de si la tarea ya se terminó de ejecutar o si se requiere más tiempo para que sea terminada), si existe un núcleo disponible entonces se busca una tarea que requiera la capacidad de ese núcleo (o menor a ésta) y se asigna, si no, se sigue el proceso sin asignar alguna tarea y se tiene que aumentar el tiempo que lleva la tarea en el núcleo. El proceso sigue y cuando una tarea es terminada, es decir cuando se tiene que el tiempo que requiere la tarea para ejecutarse y el contador que indica cuánto tiempo lleva la tarea en el núcleo son iguales, se llenan los datos de la tabla con el paso en que salió la tarea del núcleo, los cuales se asignan a otra tabla tomando en cuenta todos ellos excepto si el núcleo está disponible o no ya que no es necesario para la comprobación de resultados.

3.1. Ejemplificación de la asignación

Una vez que el proceso realiza todas las asignaciones y todas las tareas han terminado de ejecutarse se generan datos auxiliares para crear los gráficos necesarios para la visualización de los resultados.

En la figura 3.1 se muestra la asignación de las tareas en los distintos núcleos cuando se toman las decisiones teniendo los núcleos de manera ordenada creciente, es decir, se tiene que se verifica si la tarea puede ser asignada en el núcleo uno, si no, en el núcleo dos, si no, en el tres y así sucesivamente.

Por otro lado la figura 3.1 muestra las tareas asignadas a los núcleos de manera que cada vez que se verifica una tarea se tiene que los núcleos se verifican de manera aleatoria.

En ambos casos se tiene como análisis cincuenta tareas asignadas a los ocho núcleos disponibles, al realizar las comparaciones en ambos casos se observa que al tener la asignación verificada en orden de los núcleos los tiempos totales de ejecución de los núcleos no tienen un seguimiento mientras que cuando se toman de manera aleatoria el núcleo mayor es quien tiene mayor tiempo de ejecución y se tiene un decremento en los tiempos de ejecuciones totales de los núcleos menores a éste. Sin embargo, el tiempo máximo de ejecución entre ellos no obtuvo gran diferencia.

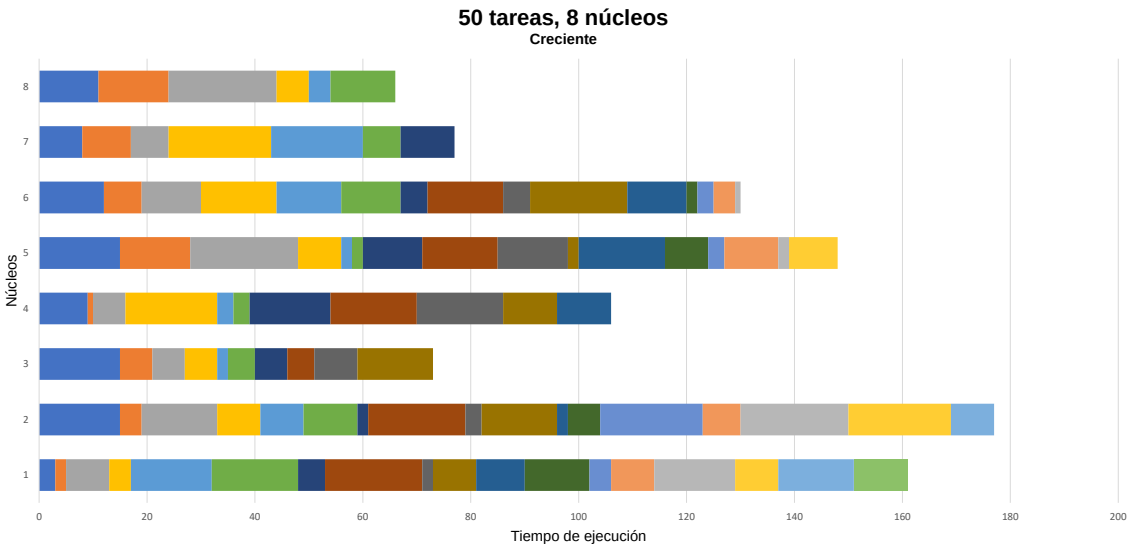

Figura 1: Ejemplificación de una asignación de tareas de manera ordenada.

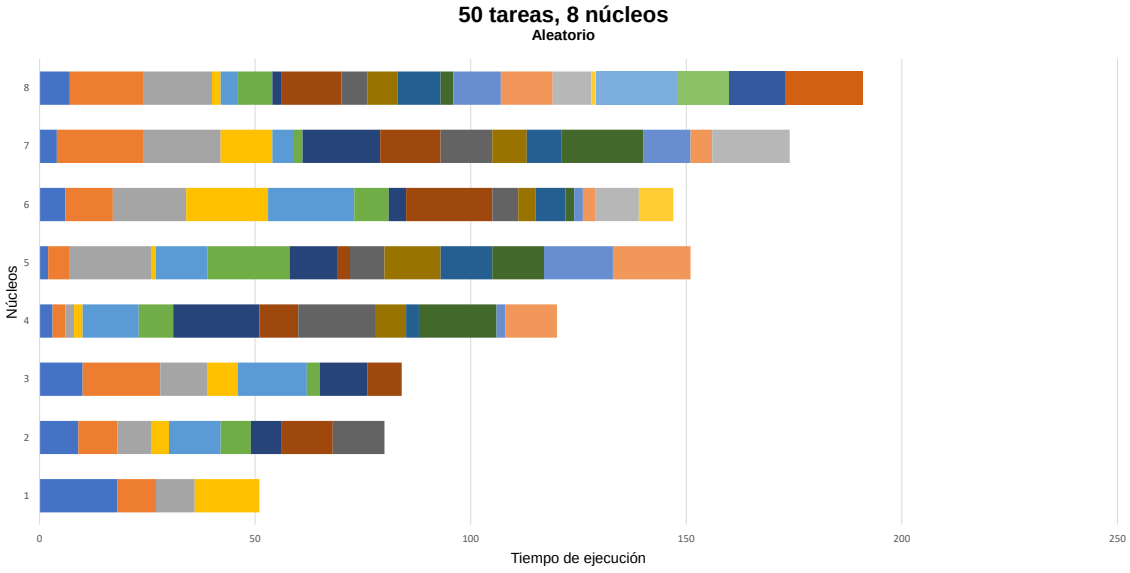

Figura 2: Ejemplificación de una asignación de tareas de manera aleatoria.

4. Resultados de la experimentación

Una vez que se realizaron las distintas repeticiones del experimento se obtuvieron gráficos similares.

En la figura 4 se observan en el nivel superior dos gráficos que indican cuántas tareas tienen como capacidad necesaria de uno a ocho núcleos (en color gris oscuro) y cuántas fueron asignadas realmente a ese núcleo (color gris claro), como se puede observar en ambas, y en el resto de gráficos de la experimentación, se tiene que no varía la cantidad tareas que requieren de ciertos núcleos en comparación con la cantidad de tareas que fueron asignadas a los distintos núcleos y los tiempos totales de ejecución, que se encuentran en el nivel inferior, se comportan de manera similar a las gráficas superiores, es decir, entre más cantidad de tareas requieren de cierto núcleo mayor va a ser su tiempo de ejecución total.

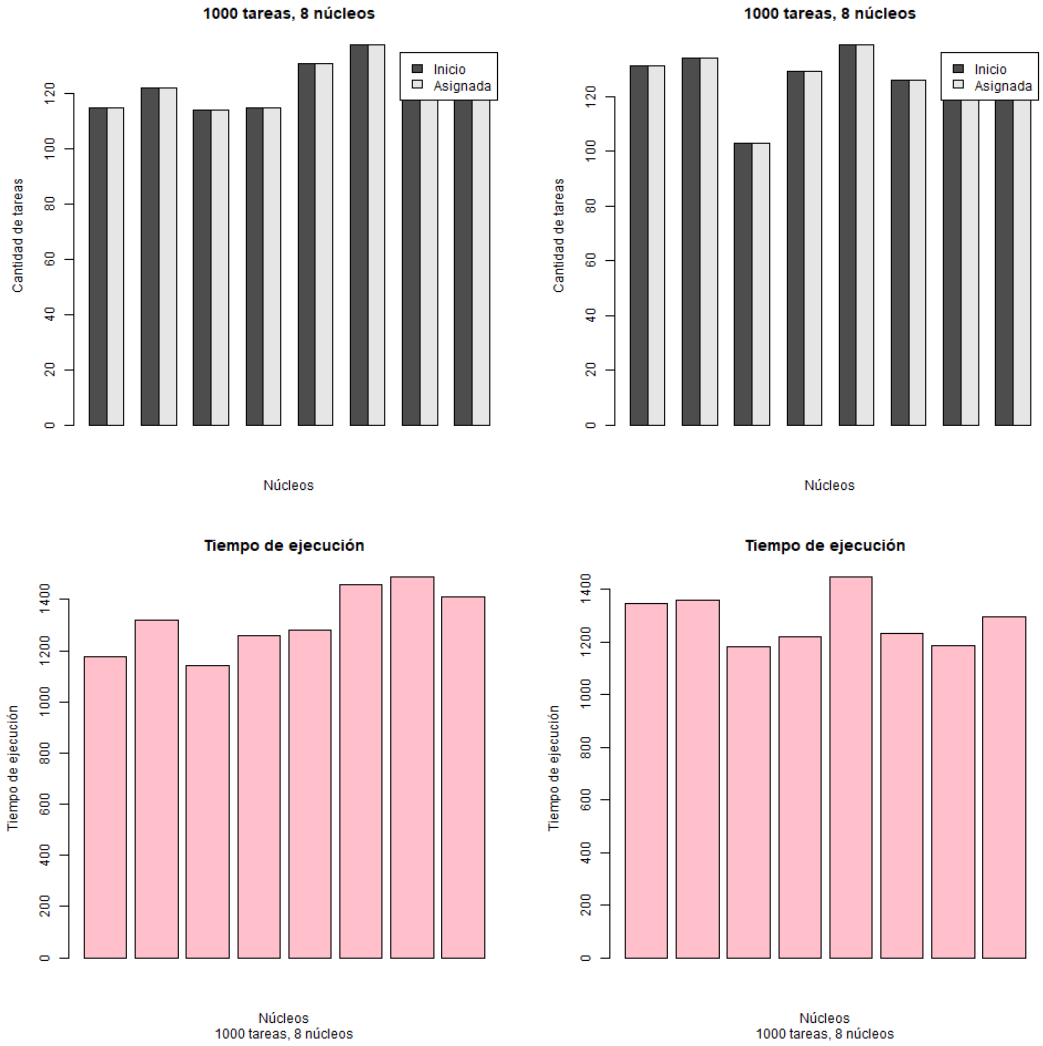

Figura 3: Comparaciones de la asignación de tareas con respecto a los tiempos de ejecución de manera ordenada.

En cambio en la figura 4 al asignar las tareas en núcleos aleatorios se observa en las gráficas superiores que la cantidad de tareas que requieren cierta capacidad de núcleo no fueron precisamente las tareas asignadas a ese núcleo.

Por otro lado, las gráficas inferiores tienen un comportamiento similar al comportamiento que se tiene con la cantidad de tareas asignadas a cierto núcleo, por lo que se puede tomar en cuenta que quien hace el efecto del tiempo total de ejecución de cada núcleo es la cantidad de tareas asignadas a él.

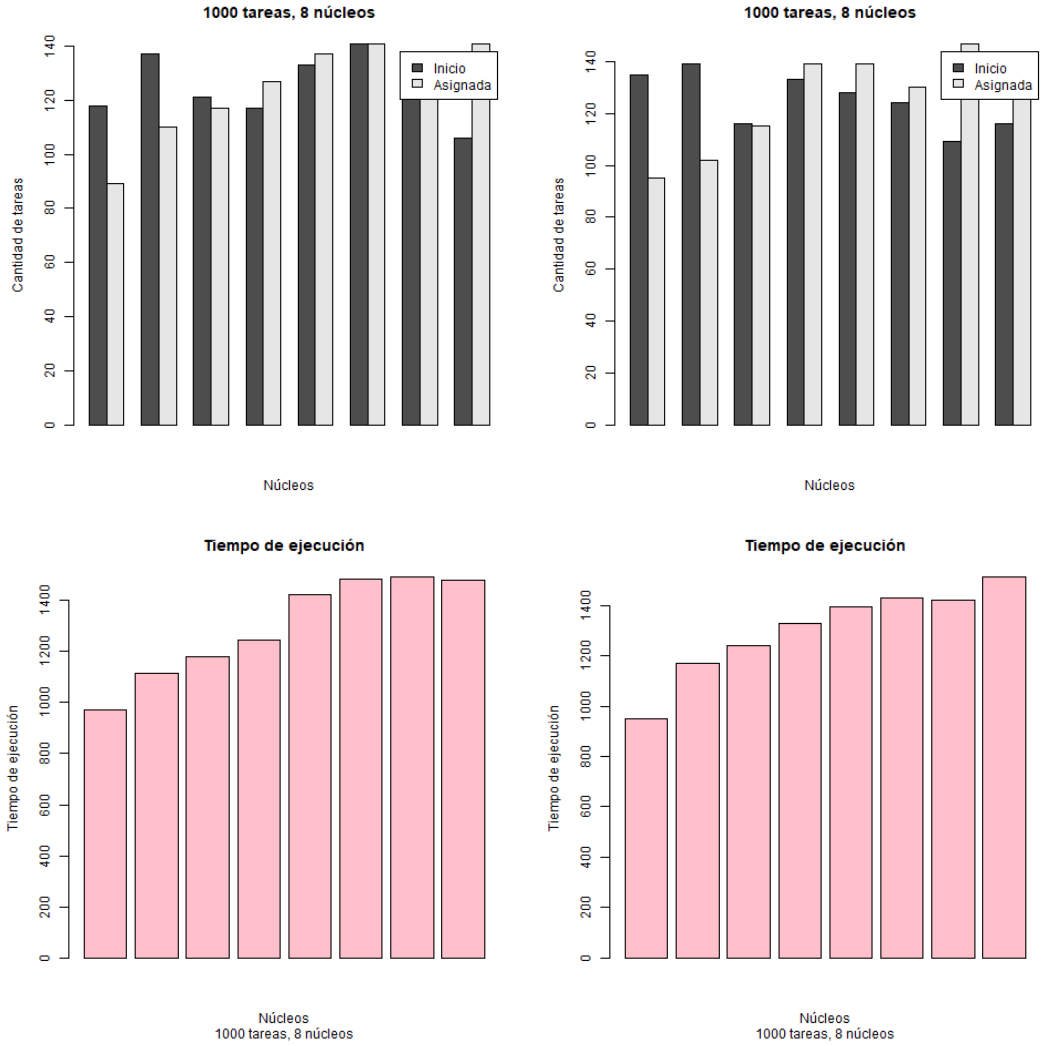

Figura 4: Comparaciones de la asignación de tareas con respecto a los tiempos de ejecución de manera aleatoria.

Al realizar las repeticiones de la simulación y obtener el tiempo máximo de ejecución de los ocho núcleos se pudo observar que los resultados son similares, todos ellos se encuentran entre mil quinientas y mil seiscientas unidades de tiempo, al ser generadas mil tareas que pueden obtener valores de tiempo de ejecución entre uno y veinte unidades de tiempo y se compararon los tiempos de manera ordenada y de manera aleatoria.

Como se observa en la figura 4 los tiempos totales de ejecución de ambos casos son similares, sin embargo se observa que en la mayoría de las repeticiones quien tiene el tiempo mínimo de ejecución es cuando la forma de tomar los núcleos es ordenada en manera ascendente, cuando el total de tareas que requerían originalmente cierto procesador se asignan en el mismo.

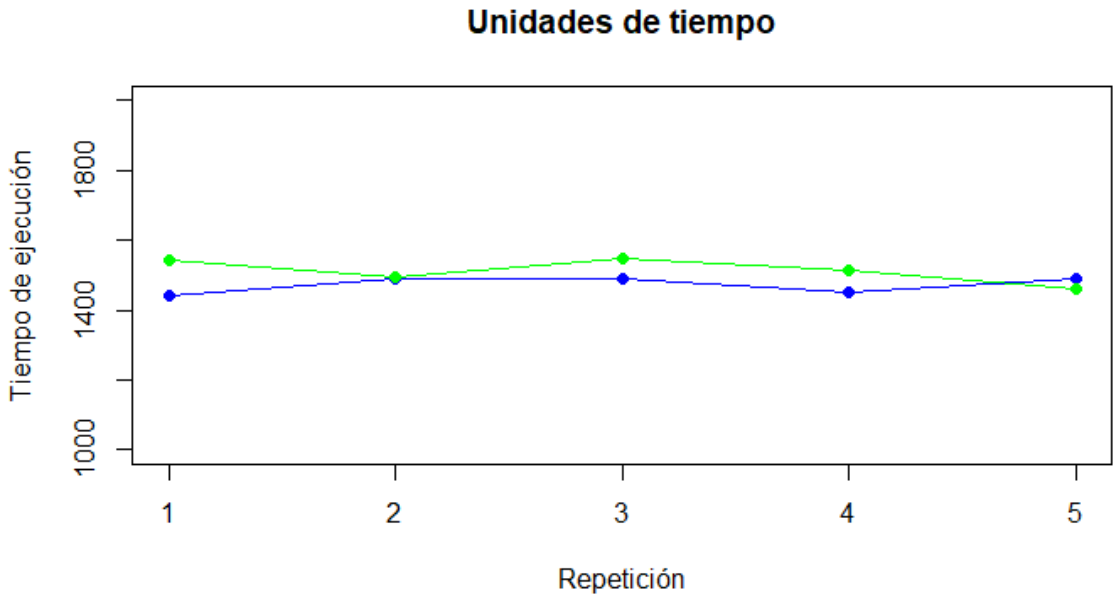

Figura 5: Tiempos máximos de ejecución de núcleos con cinco repeticiones.

5. Conclusiones

El trabajo realizado tiene una solución factible ya que lo que se busca principalmente es asignar todas las tareas que se tienen que ejecutar, sin embargo el proceso de la asignación se puede mejorar ya que al asignarlas de manera ordenada se tiene solo que exactamente la cantidad de tareas de cierto requerimiento de núcleos se asignan a ese núcleo necesario.

Como trabajo a futuro se buscará asignar las tareas de manera distinta para ver si esto ayuda en encontrar un tiempo menor de ejecución total. También se trabajará con el hecho de reducir los tiempos de espera de las tareas que se deben asignar y obtener datos que sean posibles para una experimentación concreta con un análisis estadístico.

Referencias

[1] Ezzatti P. & Nesmachnow S., "Un algoritmo evolutivo simple para el problema de asignación de tareas a procesadores"