Tarea 3: Teoría de colas

Simulación de Sistemas

28 de agosto de 2017

Introducción

La teoría de colas es un área de las matemáticas que estudia el comportamiento de líneas de espera. Los trabajos que están esperando ejecución en un cluster esencialmente forman una línea de espera. Medidas de interés que ayudan caracterizar el comportamiento de una línea de espera incluyen, por ejemplo, el tiempo total de ejecución. En esta práctica vamos a estudiar el efecto del orden de ejecución de trabajos y el número de núcleos utilizados en esta medida, examinando sí un número entero dado es un número primo.

Para esta tercer tarea se desea examinar como las diferencias en los tiempos de ejecución de los diferentes ordenamientos cambian cuando se varía el número de núcleos asignados al cluster.

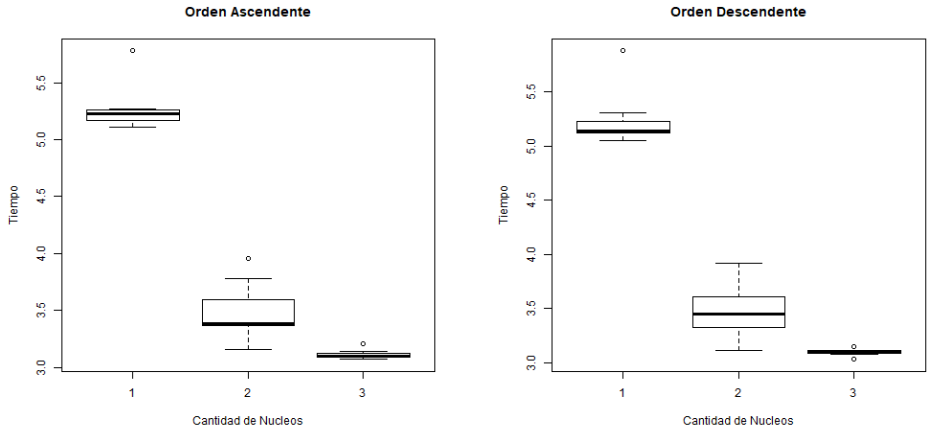
Tarea

La siguiente tarea se realizó en una máquina con las siguientes especificaciones. Procesador Intel(R)Core(TM) i5-6200U CPU 2.30 GHz 2.40 GHz con 8GB en memoria RAM y sistema operativo Windows 10 Home.

En esta práctica los ordenamientos establecidos para la serie de números eran tres: orden ascendente, orden descendente y aleatorio, además se consideraron dos ordenamientos más con la finalidad de análizar los tiempos de ejecución variando el número de núcleos asignado, dichos ordenamientos son: tomar de la serie los números primos y posicionarlos en los primeros lugares y los siguientes lugares son ocupados por el resto de los números en orden ascenente, el otro ordenamiento es muy similar la única diferencia es que los números primos son los que ocupan los últimos lugares en el ordenamento.

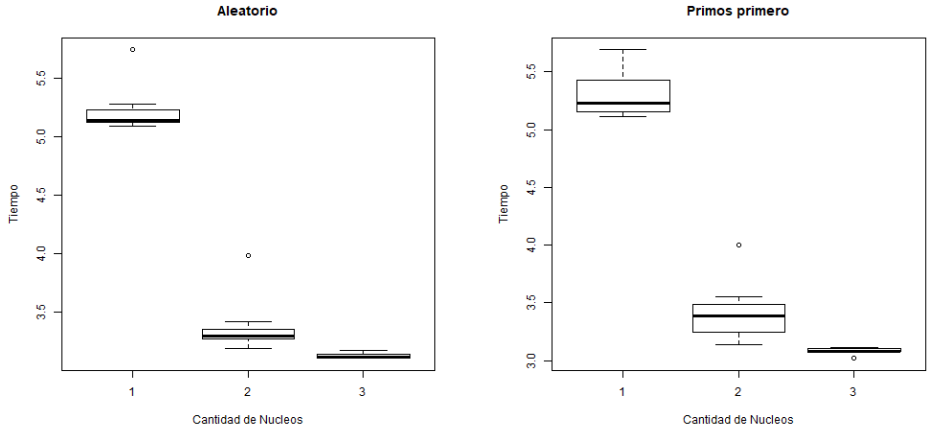
Para cada uno de los ordenamientos descritos anteriormente se corren siete replicas con uno, dos y tres núcleos para una serie de números que abarca del 1 hasta el 10000.

Los resultados obtenidos con respecto a el tiempo de ejecución se muestran en las figuras 1, 2 y 3 donde podemos notar que en cualquier ordenamiento el aumento de núcleos hace que el tiempo de ejecución disminuya drásticamente, observamos también que los ordenamientos de primos al principio al inicio y al final con un núcleo tienen peores tiempo de ejecución en comparación con lo otros ordenamentos; mientras que los tiempos más parecidos los presentaron los ordenamientos ascendente y descendente.



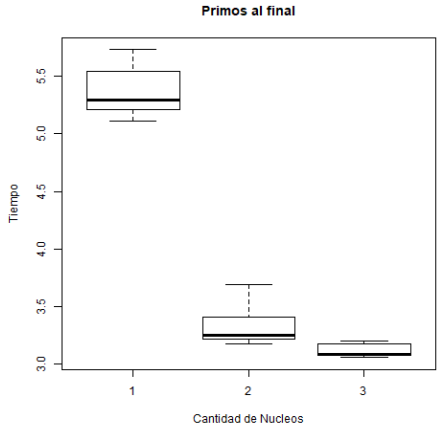
a) b)

Figura 1: Tiempos de ejecución con distinta cantidad de núcleos con serie en orden a): ascendente ; b) descendente



c) d)

Figura 2: Tiempos de ejecución con distinta cantidad de núcleos con serie en orden c): aleatoria ; d) primos al inicio



e)

Figura 3: Tiempos de ejecución con distinta cantidad de núcleos con serie en orden e): primos al final

Reto 1

La finalidad de este reto es argumentar apoyándose con visualizaciones, las posibles causas a las diferencias en los tiempos de ejecuciones y razonar cómo y por qué le afecta el número de núcleos disponibles a la diferencia.

El tiempo que toma determinar si un número n es primo depende por completo del mismo número pues esta claro que si se trata del número 1, 2 o cualquier número par el algoritmo inmediatamente detecta que no es un número primo, pero si es un número que no tiene esas características le va a tomar $\sqrt{n}$ pasos dar una solución.

Para las visualizaciones de este reto vamos a clasificar los números en tres grupos:

- Fáciles de procesar: todos los números pares y los números 1 y 2.
- Procesamiento medio: todos los números impares que no son primos.
- Díficiles de procesar: todos los números primos de la serie.

Supondremos que entre los elementos de un mismo grupo el tiempo de procesamiento es el mismo. Representaremos con verde a los números de fácil procesamiento, de color amarillo los de procesamiento medio y de rojo a los díficiles de procesar. Además consideramos que el trabajo se va asignando al primer

núcleo desocupado.

En cualquier ordenamiento cuando se tiene un solo núcleo en el cluster el tiempo total es el tiempo que toma al núcleo procesar toda la serie. Las Figuras 4, 5 y 6 nos muestran las tareas asignadas a los núcleos con orden ascendente, aleatorio y números primos al inicio para el caso de dos y tres núcleos.

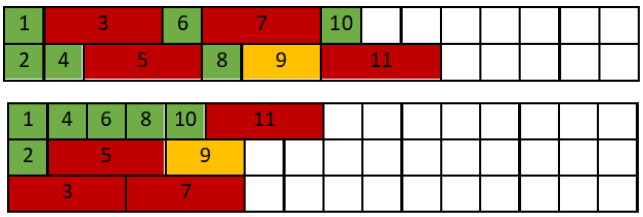


Figura 4: Orden ascendente con dos y tres núcleos

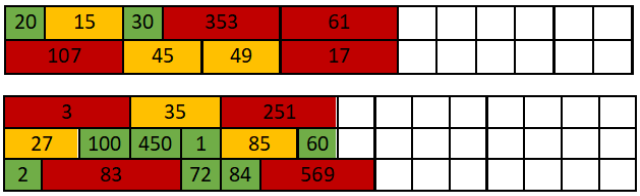


Figura 5: Orden aleatorio con dos y tres núcleos

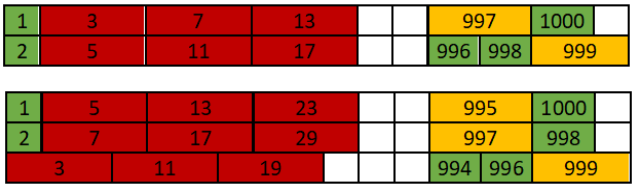


Figura 6: Orden números primos al inicio con dos y tres núcleos

Conforme aumentamos la cantidad de núcleos el tiempo de procesamento es menor para cualquier ordenamiento. Cuando el ordenamiento es el de números primos al inicio todo el trabajo pesado es procesado al inicio pero esto no cambia demasiado con el tiempo que tarda el de orden ascendente; sin embargo a mi consideración el de orden aleatorio distribuye de mejor manera el trabajo a los núcleos lo cual debería impactar en el tiempo total de ejecución.

Reto 2

El segundo reto es aplicar en R pruebas estadísticas para determinar si las diferencias observadas entre los ordenamientos son significativas.

La prueba estadística empleada es Kruskal-Wallis. Como primer objetivo deseamos determinar si hay direncias de los tiempos de ejecución y los diferentes tipos de ordenamientos. Al hacer la prueba obtenemos un p-valor mayor a 0.05 luego aceptamos la hipótesis nula de que no hay diferencias en el tiempo de ejecuciíon según el tipo de ordenamiento de la serie. Ahora bien para determinar si hay diferencias entre los tiempos de ejecución y la cantidad de núcleos asignados al cluster. La prueba realizada nos da un p-valor mucho menor a 0.05 por tanto rechazamos la hipótesis nula de que no hay diferencias en el tiempo de ejecución según la cantidad de núcleos empleados.