Modelo de urnas

Alberto Benavides

2 de octubre de 2017

Objetivos

1. Paralelizar el código encontrado en *Práctica 8: Modelo de urnas* (Shaeffer, n.d.) y medir el tiempo que se logra ahorrar.
2. Estudiar si existe ahorro estadísticamente significativo para diferentes valores de k, teniendo $n = 30k$.

Simulación y resultados

Los experimentos que se simularon en esta práctica se corrieron en una computadora portátil con sistema operativo Windows 10 Home Single Language, procesador Intel(R) Core(TM) i7-7500U CPU @ 2.70 GHz, 2904 MHz de dos núcleos principales y cuatro lógicos.

Del código compartido por Shaeffer (n.d.) se paralelizaron las funciones `romperse`, `unirse` y una función sin nombre que une las urnas que están listas para hacerlo después de pasar por la función `unirse`. Esta paralelización se llevó a cabo por la función `parSapply` incluida en la librería `parallel` del lenguaje R. Las dos primeras funciones se corrieron para cada una de las filas de las tablas de frecuencias donde se agruparon las urnas por tamaños.

Los resultados obtenidos se devuelven, para cada caso, en forma de lista, por lo que hay que convertirlos a vectores numéricos con el fin de agruparlos nuevamente como tamaños de urnas. Esto se logró mediante la función `unlist` a la que se le pasó como parámetro la variable `cumulos` donde se almacenaban enlistados los resultados.

```
cumulos <- parSapply(cluster, 1:nrow(freq), romperse)
cumulos <- unlist(cumulos)
[...]

cumulos <- parSapply(cluster, 1:nrow(freq), unirse)
cumulos <- unlist(cumulos)
```

La última función paralelizada, en la que se agregan entre sí y por parejas los cúmulos listos para agruparse almacenados en `juntarse`, suma los grupos de números previamente seleccionados por la función `unirse` y desordenados aleatoriamente.

```
cumulosUnidos <- parSapply(cluster, 1:floor(nt / 2),
  function(i){
      return(juntarse[2*i-1] + juntarse[2*i])
  }
)
```

Todo el código de la práctica, con los cambios mencionados, se encapsuló en una función denominada `experiment`, ésta se corrió de uno a cuatro núcleos con una duración de cincuenta pasos para medir las diferencias de tiempo de ejecución con base en el uso de núcleos. La imagen 1 (p. 3) muestra los resultados descritos y que el menor tiempo de ejecución corresponde con el uso de dos núcleos.

Posteriormente se iteraron, dado $p = [3, 6]$, la cantidad de cúmulos, $k = 10^p$, y el total de partículas, $n = 3k$. Con estas nuevas condiciones se corrió el experimento para uno y dos núcleos puesto que dos núcleos fue el número de núcleos que logró el tiempo óptimo mínimo para el experimento, y un núcleo corresponde a corridas no paralelizadas. Los tiempos, en segundos, por k inicial se graficaron en escala logarítmica en la figura 2 (p. **??**Paralelizar)).

Al obtener la matriz de coeficientes de correlación se tiene una $r_1 = -0.028$ para la correlación entre el tiempo y los núcleos, mientras que obtenemos una $r_2 = 0.998$ entre el tiempo requerido y la cantidad de cúmulos inicial.

Conclusiones

1. Paralelizar reduce el tiempo de ejecución de funciones susceptibles de paralelización, pero el tiempo mínimo se logra al utilizar sólo los núcleos físicos disponibles con respecto al uso de núcleos lógicos.
2. Se puede que como r_1 es muy cercano a cero, existe un ahorro mínimo de tiempo conforme crece el número de núcleos al variar el número de cúmulos inicial, el cual es estadísticamente no significativo.

Referencias

Shaeffer, Elisa. n.d. *Práctica 8: Modelo de Urnas*. http://elisa.dyndns-web.com/teaching/comp/par/p8.html.

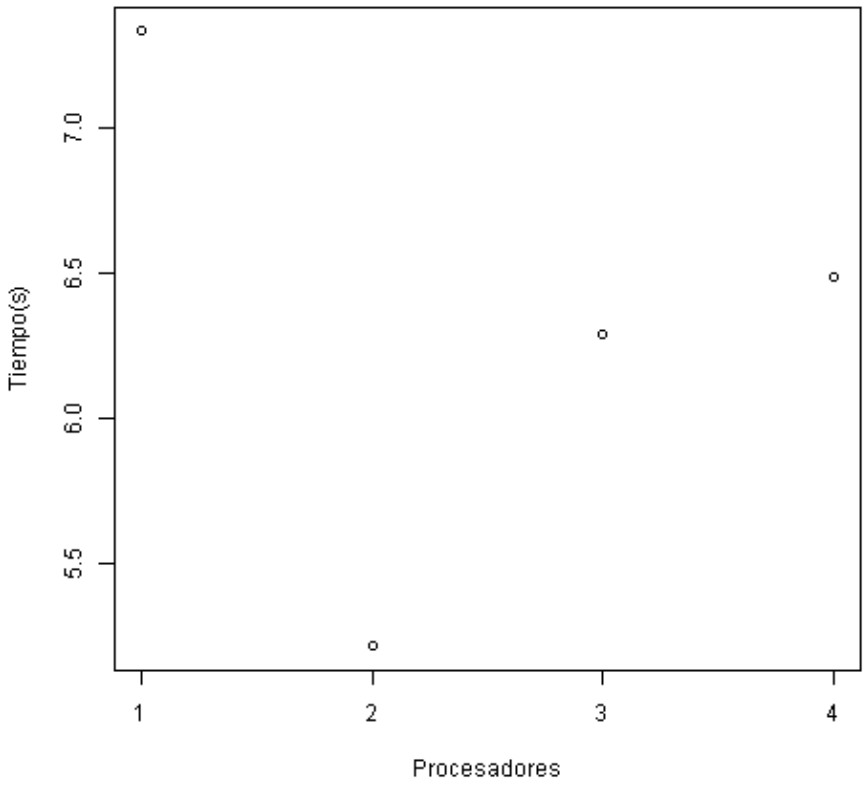

Figura 1: Tiempo en segundos que tomó correr este experimento con los núcleos disponibles en el equipo descrito.

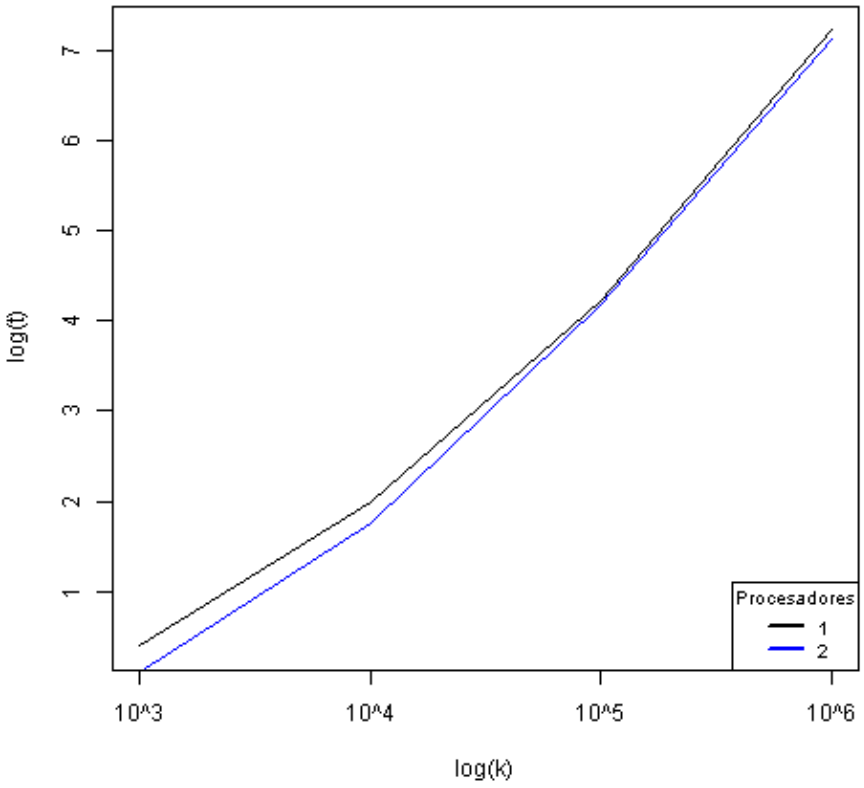

Figura 2: Comparación de tiempos con k iniciales medidos en escala logarítmica para uno (no paralelizado) y dos (paralelizado) núcleos.