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 en comparación con el código secuencial.
2. Estudiar si existe ahorro estadísticamente significativo para diferentes valores de cúmulos y cantidad de partículas iniciales.

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.

En este experimento se parte de una cantidad k de cúmulos que pueden contener n partículas en total. Estos cúmulos serán susceptibles de separarse y unirse durante cincuenta iteraciones con probabilidades sigmoidal y exponencial respectivamente, basadas en sus medianas como puntos críticos. Una representación visual de los cúmulos como círculos de radio igual al tamaño de sus partículas puede consultarse en la figura 1 (p. 2) donde se grafican los cúmulos en posiciones aleatorias en un plano cartesiano. Adicionalmente, el desarrollo de separación y unión de cúmulos como barras de frecuencia puede consultarse en https://goo.gl/B2TAAR.

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

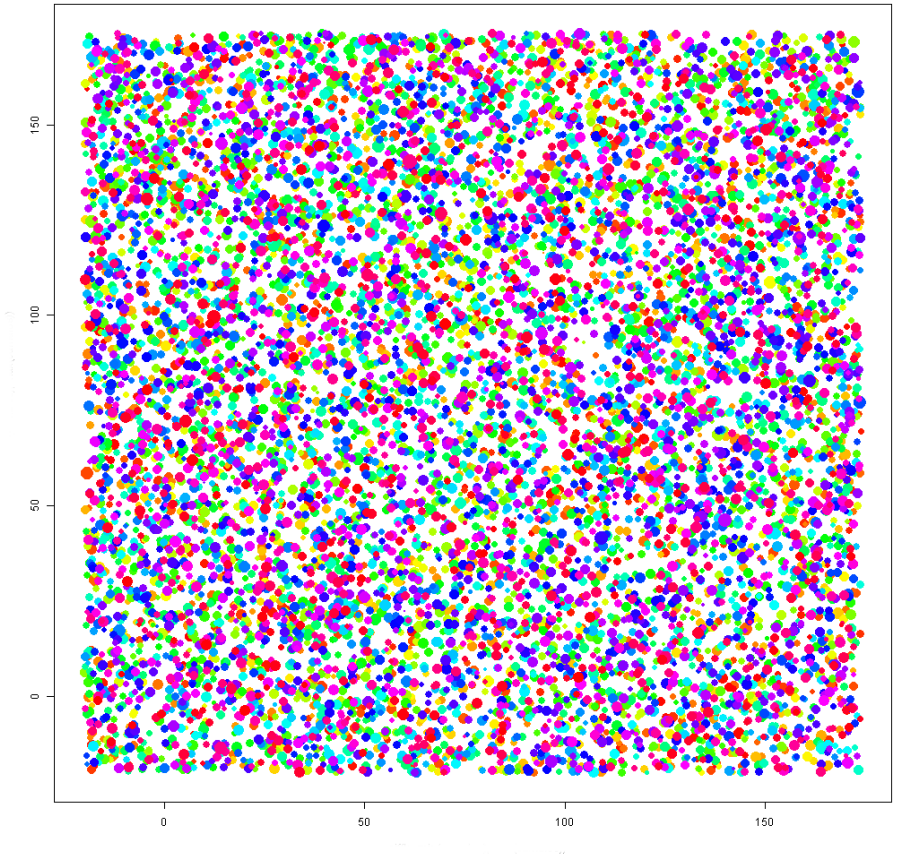

Figura 1: Representación en posiciones aleatorias de los cúmulos como círculos de radio igual al tamaño de partículas con que están constituidos.

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 almacenaron 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 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 que se corrió de uno a cuatro núcleos para medir las diferencias de tiempo de ejecución con base en el uso de núcleos. La imagen 2 (p. 4) muestra los resultados descritos y que el menor tiempo de ejecución corresponde con el uso de dos núcleos.

Posteriormente se iteraron en un rango $p = [3, 6]$ 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 mínimo para el experimento, y un núcleo corresponde a corridas no paralelizadas. Los tiempos en segundos por número de cúmulos inicial se graficaron en escala logarítmica en la figura 3 (p. 5).

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 se obtiene una $r_2 = 0.998$ entre el tiempo requerido y la cantidad de cúmulos inicial.

Conclusiones

1. Evidentemente, paralelizar reduce los tiempos de ejecución de funciones que pueden paralelizarse, pero el tiempo mínimo no se logra al utilizar todos los núcleos disponibles con independencia de que sean físicos o lógicos, sino que el mejor tiempo se tiene cuando se utilizan sólo los núcleos físicos disponibles.

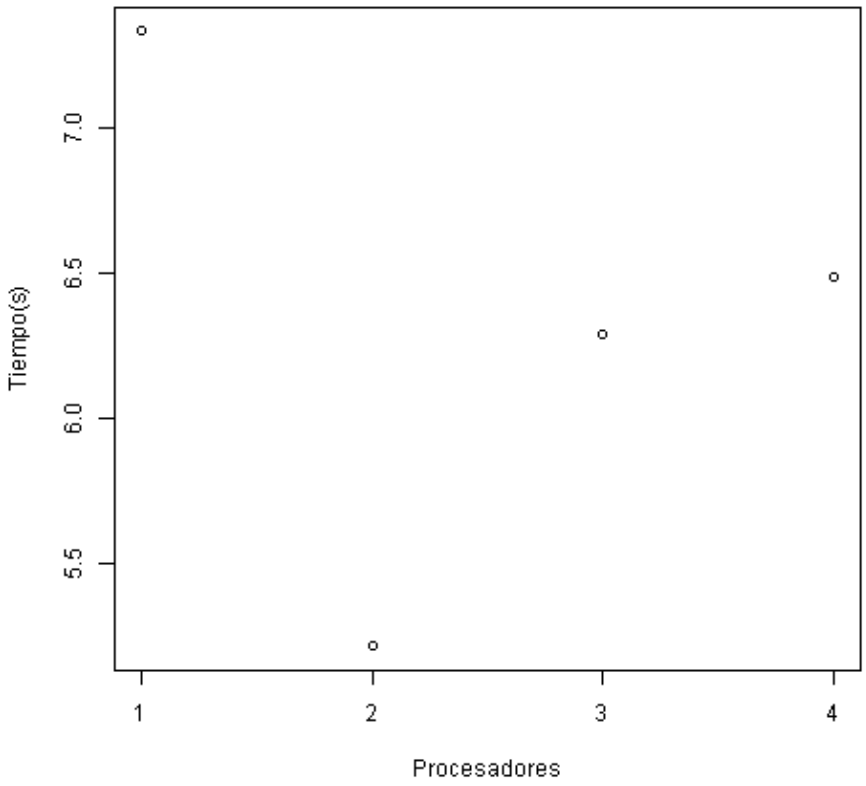

Figura 2: Tiempo en segundos que tomó correr el experimento con los núcleos disponibles en el equipo descrito.

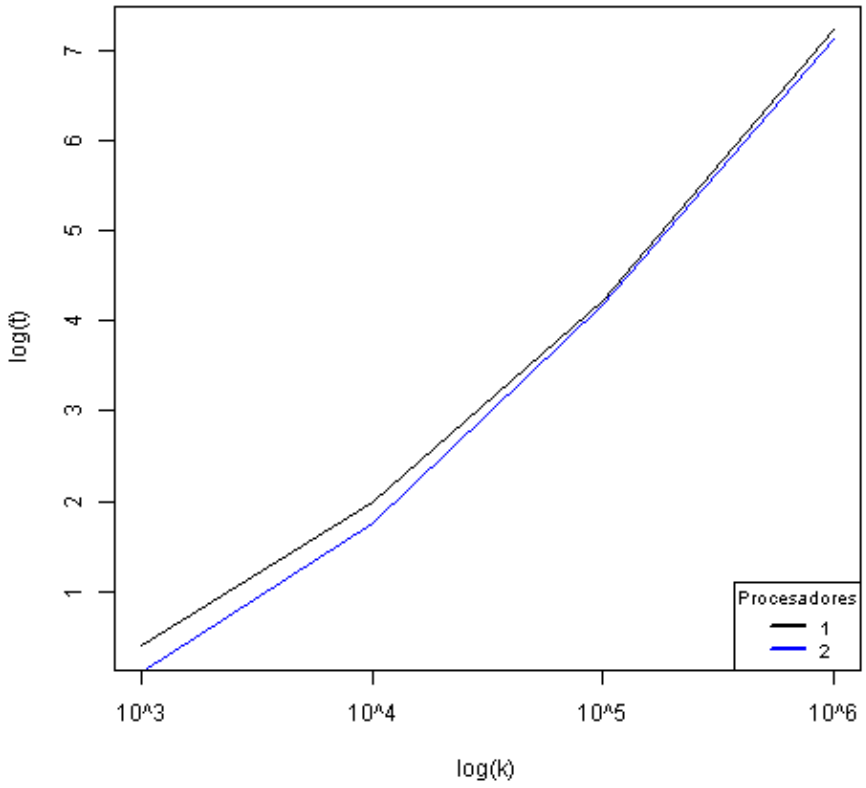

Figura 3: Comparación de tiempos con k iniciales medidos en escala logarítmica para uno (no paralelizado) y dos (paralelizado) núcleos.

2. Puesto que r_1 es negativo y muy cercano a cero, existe un ahorro mínimo de tiempo conforme crece el número de núcleos al variar el número de cúmulos inicial.

Referencias

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