Práctica 5: método Monte-Carlo

19 de septiembre de 2017

1. Introducción

El objetivo de ésta prueba consiste en implementar el método Monte-Carlo en la aproximación del área bajo una curva, así mismo variar los parámetros de la prueba con el fin de observar si alguna variación ofrece una mayor precisión al resultado. Se comparó el resultado obtenido con el resultado que ofrece *Wolfram Alpha* (WA) con los mismos parámetros.

2. Parámetros de trabajo

La experimentación se realizó en una iMac con procesador Intel(R)Core(TM) i5-6200U CPU 2.30 GHz 2.40 GHz con 8 GB en memoria RAM y sistema operativo Windows 10 Home.

Se establecieron los tamaños de muestra con un crecimiento exponencial, ya que cuando el aumento de puntos era de forma lineal la diferencia era muy poca y no se podía apreciar el efecto de los cambios. Para $n \in \{2^{10}, 2^{12}, 2^{14}, 2^{16}, 2^{18}, 2^{20}, 2^{22}\}$ con una reptición de veinte puntos, de lo cual cada punto representa el promedio de 200 aproximaciones, para los valores del intervalo de la integral fueron de $x \in [3, 7]$

$$\int_{x=3}^{x=7} \frac{1}{e^x + e^{-x}} dx, \tag{1}$$

y el valor obtenido de la plataforma WA con el cual se comparó el error fue 4.8834×10^{-3}.

3. Modificaciones del código

Se creó una función para calcular el error o la diferencia entre los valores obtenidos con el registrado por WA, así mismo se crearon clusters con el fin de poder paralelizar lo más que se pudiera, utilizando siete de los núcleos de la máquina con el fin de que no perdiera su funcionalidad utilizandolos todos. Así mismo se midieron los tiempos de corrida para cada parámetro con la función *system.time* y se crearon *dataframes* para guardar cada valor obtenido.

```
fcontagios <- function(agentes,i){
contagios <- rep(FALSE, n)
for (i in 1:n) { # posibles contagios
a1 <- agentes[i, ]
if (a1$estado == "I") { # desde los infectados
for (j in 1:n) {
if (!contagios[j]) { # aun sin contagio
a2 <- agentes[j, ]
if (a2$estado == "S") { # hacia los susceptibles
dx <- a1$x - a2$x
dy <- a1$y - a2$y
d <- sqrt(dx^2 + dy^2)
if (d < r) { # umbral
p <- (r - d) / r
if (runif(1) < p) {
contagios[j] <- TRUE
} } } } } } }
```

Figura 1: Aproximaciones al valor obtenido por WA, el cual es representado por la linea recta, para las diferentes cantidades de puntos.

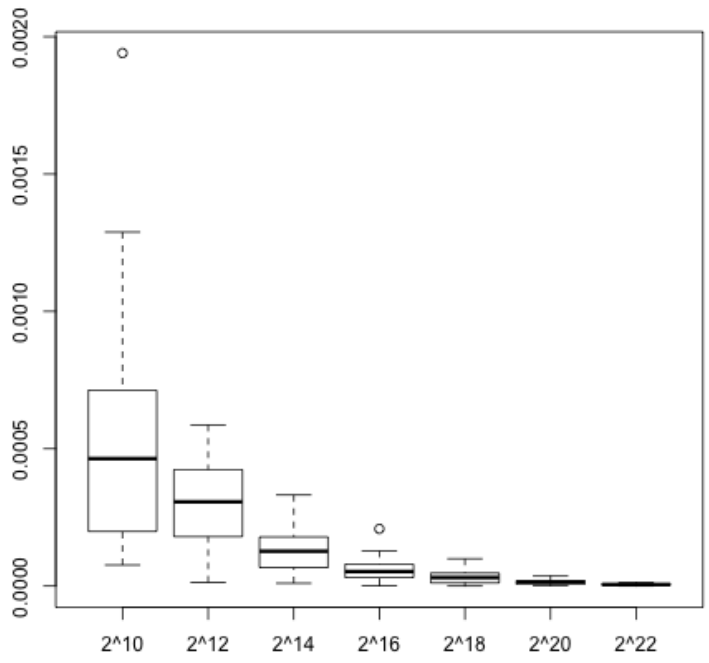

Figura 2: El error o diferencia obtenido para cada valor de puntos respecto al obtenido con WA.

```
return(contagios)
}
```

4. Resultados

Se muestran a continuación figura 1, 2 y 3 las cuales reflejan de un modo gráfico los comparativos y los resultados de la experimentación, para cada una de ellas se utilizó la escala que mejór reflejara los valores para todos los casos.

4.1. Interpretación

En la figura 2 se observa que para los primeros valores de puntos, los cuales eran muy bajos, la aproximación del valor de el área era muy disperso, e incluso aunque sí se aproximaba al valor dado por WA la mayoría de los puntos no lo hacían, en cambio, al crecer la cantidad de puntos más allá del 2^{16} la disperción de los valores se reduce, así mismo la aproximación tiene una mayor precisión.

En la figura 2 podemos reafirmar lo que se dijo de la figura 1, ya que en ésta figura representa el error que se tiene respecto al valor de WA, es decir, el valor absoluto de la diferencia de estas dos cifras. De igual modo se observa una notoria disminución del error a partir de 2^{16}, puntualizando que ésta casi llega al cero en el valor de 2^{22} en la mayoría de sus réplicas.

Comportamiento contrastante y como es de esperarse en la figura 3 podemos observar el modo de aumento en tiempos de ejecución para cada uno de los valores de puntos, el cual es medido en segundos. Hasta el valor de 2^{16} el tiempo de ejecución es relativamente pequeño, y podemos observar que no varía mucho de una replica a otra, pero al momento de llegar al valor de 2^{18} es cuando podemos observar el crecimiento de una manera más notoria.

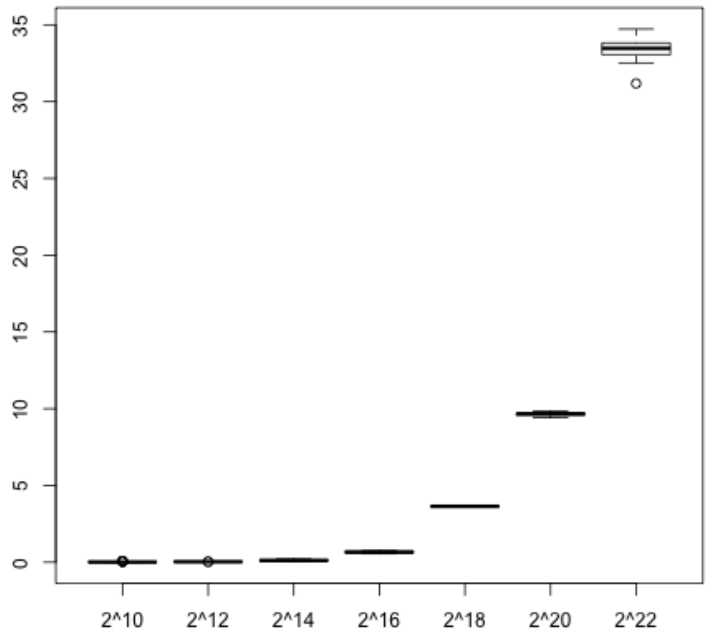

Figura 3: El tiempo de computo para cada uno de los casos.

Es decir, podemos aumentar el número de puntos y réplicas lo más que nuestro ordenador lo permita y seguro que nuestra precisión aumentará, pero es de considerar de igual forma que los tiempos de ejecución se dispararán, en este caso también se intentaron correr las réplicas para los valores de $2^{24}, 2^{26}, 2^{28}$ pero la máquina se quedaba mucho tiempo procesando en la primera de estas tres cifras e interrumía la ejecución por falta de memoria.

5. Reto 1

Para el reto uno se tomó la estimación del valor de ϕ de Kurt y se implementó paralelismo, así mismo fueron realizadas las mismas comparaciones que en la prueba anterior, es decir, los parámetro fueron variados y fue observado lo que ocurre con la aproximación, el error y los tiempos de ejecución.