Práctica 6

Sistema multiagente

Introducción

En esta práctica se simula la propagación de una epidemia en una población (agentes), estos agentes pueden tomar tres estados los cuales son susceptibles (S), infectados (I) o recuperados (R). Las variables a considerar en esta práctica son: la probabilidad de infección inicial y el número de agentes.

Objetivos

1. Evaluar las partes del código original para determinar que partes de éste se pueden paralelizar y disminuir el tiempo de ejecución.
2. Reto 1. Determinar el efecto en porcentaje de infectados, si al inicio de la simulación se vacunan con una probabilidad *pv* una cantidad de agentes.
3. Reto 2. Determinar el efecto de la probabilidad de infectados iniciales contra el porcentaje de infectados máximo.

Simulación y resultados

Se evaluó el código original para determinar las partes que fueran efectivas paralelizar y así ahorrar tiempo en la ejecución del mismo. Como resultado de esta evaluación se consideró paralelizar la parte de contagios del programa original. En esta parte del programa se realiza la evaluación de nuestro data frame agentes que es donde se encuentra presentada la población inicial y cuenta con las columnas de *x*, *y* para indicar la posición actual de cada agente, Δx y Δy que determinan el movimiento de cada agente y una columna estado que nos indica el estado en el que se encuentra dicho agente. La parte de contagios se encarga de comprobar paso por paso de la simulación quienes son los infectados y a quienes estos infectados pueden contagiar para que se puedan considerar nuevos infectado en el paso siguiente. Se determinó que esta parte del código es importante de paralelizar ya que para cada agente infectado es necesario calcular la distancia euclidiana con todos los demás agentes y determinar si se encuentran dentro del umbral de contagio, como el cálculo de la distancia para comparar a cuantos puede contagiar el agente infectado en cuestión es independiente para cada agente, esta parte se puede paralelizar ya que el cálculo de la distancia se puede realizar para todos los agentes al mismo tiempo y de esta manera ahorrar tiempo en la simulación

Para realizar esta paralelización se modificó la parte de contagios del código original y se convirtió en la función contagiados, en esta función lo que se hizo fue el mismo calculo que se realiza en el código original y al final la función devuelve los índices de los nuevos agentes contagiados y que estarán infectados en el siguiente paso.

```
contagiados<- function (j){
    contagios=rep(FALSE,n)
    for (i in 1:n) {
        if (agentes[i,5] == "I") { # desde los infectados
            if (!contagios[j]) { # aun sin contagio
                #a2 <- agentes[j, ]
                if (agentes[j,5] == "S") { # hacia los
  susceptibles
                    dx <- agentes[i,1] - agentes[j,1]
                    dy <- agentes[i,2] - agentes[j,2]
                          d <- sqrt(dx^2 + dy^2)
                          if (d < r) { # umbral
                              p <- (r - d) / r
                              if (runif(1) < p) {
                                  contagios[j]<-TRUE
                              }
                          }
                      }
                  }
              }
          }
          return (contagios[j])
      }
```

Posteriormente esta función se utilizó con el paquete doParallel específicamente con el foreach como se muestra a continuación

```
       #paralelizar contagiados
      contagiios=foreach(j=1:n,
  .combine = c) %dopar% contagiados(j)
          stopimplicitCluster()
```

Para realizar la comparación de los tiempos de ejecución entre el programa original y el paralelizado, se generó un nuevo código para mandar a llamar con la función source cada uno de los programas y para cada uno se realizaron 10 réplicas con un número de agentes de 100 y una probabilidad de infección de 0.05 Los tiempos obtenidos se pueden verificar en la figura 1.

```
x=data.frame()
Resultados=data.frame()
for (i in 1:10){

  source('~/GitHub/SimulacionComputacional/P6/p6paralelizado/p6.
  R', encoding = 'UTF-8')

  source('~/GitHub/SimulacionComputacional/P6/original/original.
  R', encoding = 'UTF-8')
  Resultados=cbind(TiempoT,TiempoO)
  x=rbind(x,Resultados)
      }

      colnames(x)=c("Programa Paralelizado", "Programa Original")
      png("Prac6.png",width=600, height=800,pointsize = 20)
      boxplot(x,col=c("Blue","Red"),ylab="Tiempo de ejecución
  (s)")
      graphics.off()
```

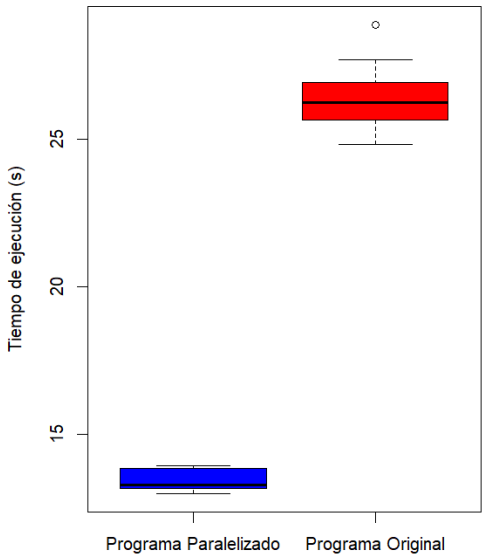


Figura 1.Caja-bigote de la comparación de los tiempos de ejecución

Como se puede observar para un número de 100 agentes la diferencia entre el tiempo de ejecución para el programa paralelizado contra el original es del doble. Esto quiere decir que el programa paralelizado es el doble de rápido que el programa original. Cabe mencionar que para una cantidad de agentes de 50 la diferencia entre los programas es mínima. Sin embargo, al aumentar la cantidad de agentes es cuando más brilla el programa paralelizado.

Reto 1

Para primer reto se hicieron las modificaciones en la parte de la creación del data frame agentes para que antes de comenzar a infectar existiera la probabilidad de que el inicio de la simulación se vacunaran a los agentes con una probabilidad *pv* y esta se fue variando para observar el efecto en el máximo de infectados.

```
1. agentes <- data.frame(x = double(), y = double(), dx = double(),
   dy = double(), estado  = character())
2. for (i in 1:n) {
3.   e <- "S"
4.   if (runif(1)<pv){e<-"R"}
5.   if (runif(1) < pi && e!="R") {
6.     e <- "I"
7.   }
8.   agentes <- rbind(agentes, data.frame(x = runif(1, 0, l),
   y = runif(1, 0, l),
9.                                        dx = runif(1, -v, v),
   dy = runif(1, -v, v),
10.                                         estado = e))
11.        levels(agentes$estado) <- c("S", "I", "R")
12.      }
```

En la figura 2 se puede observar el primer paso de la simulación y comprobar que en efecto existen agentes con el estado de recuperado (R).

Los resultados y el efecto del aumento en *pv* se pueden observar en la figura 2. Se observa que al aumentar el porcentaje de agentes vacunados al inicio. El número máximo de

infectados se desplaza hacia la derecha del grafico esto quiere decir que al tener mayor cantidad de vacunados en un inicio el punto máximo de infectados tarda más pasos en alcanzarse. Sin embargo, a pesar de que el porcentaje va en aumento en cada gráfico, si lo comparamos la cantidad de porcentaje para cada gráfico vemos que el porcentaje de infectados disminuye de manera directamente proporcional a la probabilidad de vacunación inicial (*pv*).

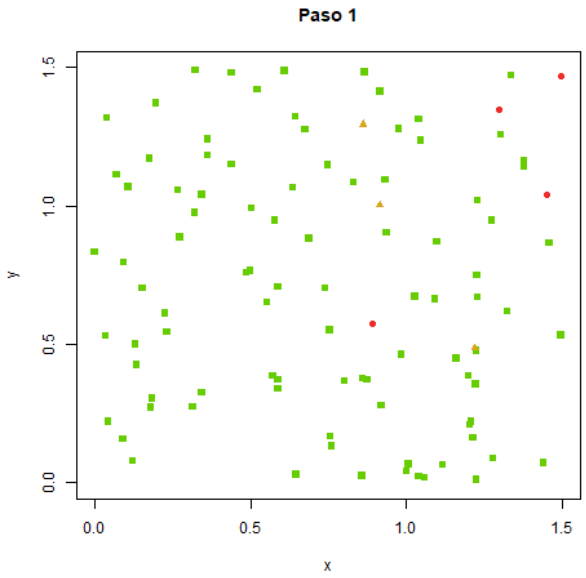
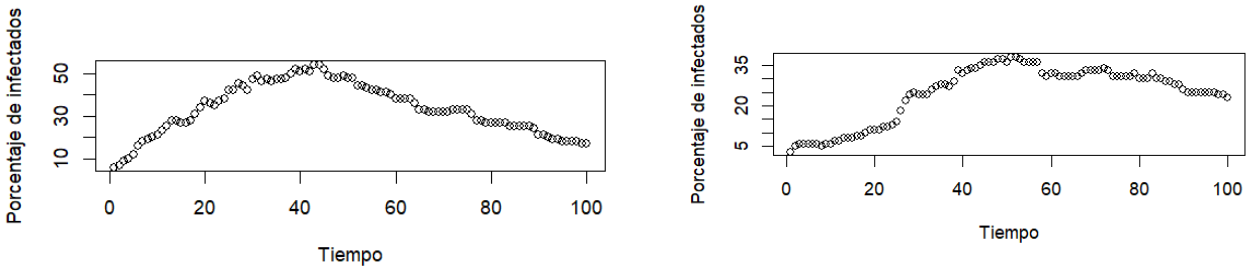
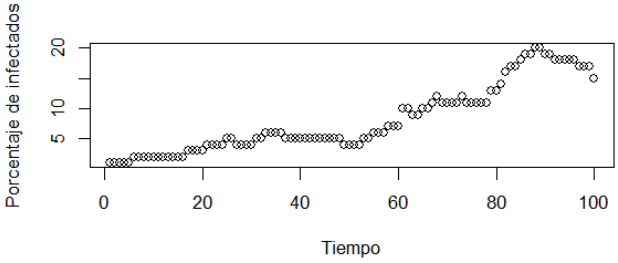


Figura 2. Ilustración del primer paso de la simulación del sistema multiagentes para un pv = 0.02.



a) b)



c)

Figura 3. Gráfico de porcentaje de infectados contra tiempo a) *pv* = 0.02, b) *pv* = 0.3, c) *pv* = 0.5

Reto 2

Para el segundo reto se modificó el código para variar la probabilidad de infectados incial (*pi*) esto se logró con ayuda de un for al inicio que fuera cambiando dicha variable el código quedó de la siguiente manera:

```
for (pi in c(0.05,0.1,0.5,0.8,1)){
  agentes <- data.frame(x = double(), y = double(),
dx = double(), dy = double(), estado  = character())
for (i in 1:n) {
  e <- "S"
  if (runif(1) < pi) {
    e <- "I"
  }
  agentes <- rbind(agentes, data.frame(x = runif(1, 0, l),
y = runif(1, 0, l),
                                       dx = runif(1, -v, v),
dy = runif(1, -v, v),
                                            estado = e))
      levels(agentes$estado) <- c("S", "I", "R")
    }
```

En la figura 4, se puede observar que el porcentaje máximo de infectados aumenta al aumentar la probabilidad de infección inicial (*pi*), el porcentaje máximo de infectados va aumentando de manera directamente proporcional. Esto es debido a que al tener una mayor cantidad de agentes infectados el inicio esto ocasiona que la propagación del a epidemia sea más rápida y haya mayor probabilidad de que un agente susceptible se encuentre con uno que esté infectado.

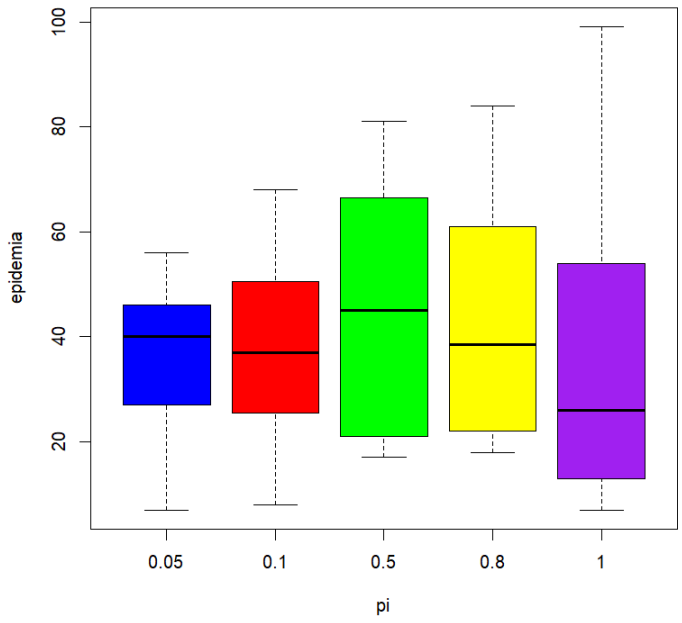


Figura 4. Caja-Bigotes la comparación del porcentaje de infectados contra la probabilidad inicial de infectados.

Conclusiones

Se determinó que el programa era paralelizable y era posible mejorar el tiempo de ejecución, pero donde más se notaba eran cuando el número de agentes estaba por encima de 50. El valor de la probabilidad de infectados inicial es directamente proporcional al porcentaje máximo de infectados en una simulación. Vacunar con una probabilidad *pv* a los agentes antes de comenzar a infectar se verá reflejado en un decremento del porcentaje máximo de infectados y corriendo en el número de pasos para el mismo.