Practica 2: Automatas celulares

José Daniel Mosquera-Artamonov

August 21, 2017

1 Introduction

El presente documento presenta el reporte de la practica de los automatas celulares realizado en la clase de Simulación de sistemas.

1.1 Especificaciones computacionales

La presente practica se realizó en una computadora con procesador Intel(R) Xeon (R) CPU E3-1245 v3 @3.4GHz 3.4GHz con una capacidad de 16 GB en memoria RAM y 8 nucleos.

1.2 Especificaciones experimentales

Se considera un rango de exploración de 0 a 1 con incrementos de 0.05 para la proporción de automatas vivos en la cuadricula. Se inspecciona a lo maximo 20 iteraciones para determinar si el automata murio o no. Cada experimento genera 30 replicas con la misma configuración. Se probaron tres dimensiones 10,25,50. Se consideran cuatro nucleos para los retos 1 y 2 como información inicial, como probabilidad de expansión se considero 10%. El maximo numero de iteracciones permitido para el reto 1 es : $dim + entero(dim * 50\%)$

2 Numero de iteracciones maximo con vida

En está sección se presenta el analisis realizado a la cantidad maxima de iteraciones que el automata dura sin morir[1]. El codigo realizado genera las graficas automaticamente y las guarda en formato pdf, llamandolas P2_N_D (N=normal, D=dimension)

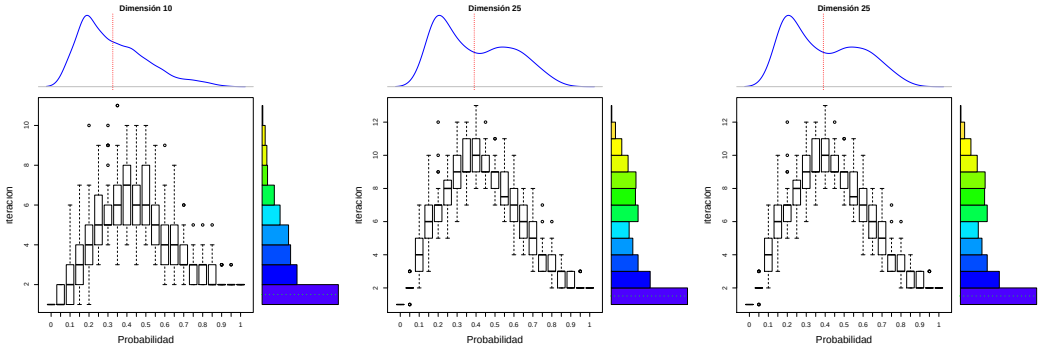
En la figura 1, se presenta el codigo modificado usado para la experimentación de está sección.

```
aq<-data.frame()
for ( dim in dim){
  clusterExport(cluster, "dim")
for (vivos  in  probavivo){
  clusterExport(cluster, "vivos")
  resultados = numeric()
  for (replica in 1:30) {
    actual <- matrix(1*(runif(num)<vivos), nrow=dim, ncol=dim)
    for (iteracion in 1:20) {
      clusterExport(cluster, "actual")
      siguiente <- parSapply(cluster, 1:num, paso)
      if (sum(siguiente) == 0) { # todos murieron
         resultados = c(resultados, iteracion, vivos)
         aq<-rbind(aq,c(dim,replica,vivos,iteracion))
         break;
      }
      actual <- matrix(siguiente, nrow=dim, ncol=dim, byrow=TRUE)
    }
  }
}
}

stopCluster(cluster)
colnames(aq)<-c("dimension","Replica","Probabilidad", "iteracion")
```

Figure 1: Codigo modificado

En la figura 2, se muestra como va variando la distribucción tanto de probabilidad en elementos vivos como el numero ade iteracciones con diferentes dimensiones en la cuadricula. a Cada uno de los subgraficos 7a 7b y 2c se le agregado densidad de probabilidad para los agentes antes de morir con su respectiva media (color rojo), ademas de graficas de barras para determinar en que iteracción la mayoria de los automatas se mueren.

[1]El codigo fuente de está seccion se ha nombrado "Practica_2.1.R"



(a) 10*10 (b) 25*25 (c) 50*50

Figure 2: Múltiples imágenes

3 Distribución de los automatas que llegan al borde o no

En está sección se presenta el analisis realizado para el primer reto que simula el crecimiento de un fractura, una epidemia entre otros casos[2].

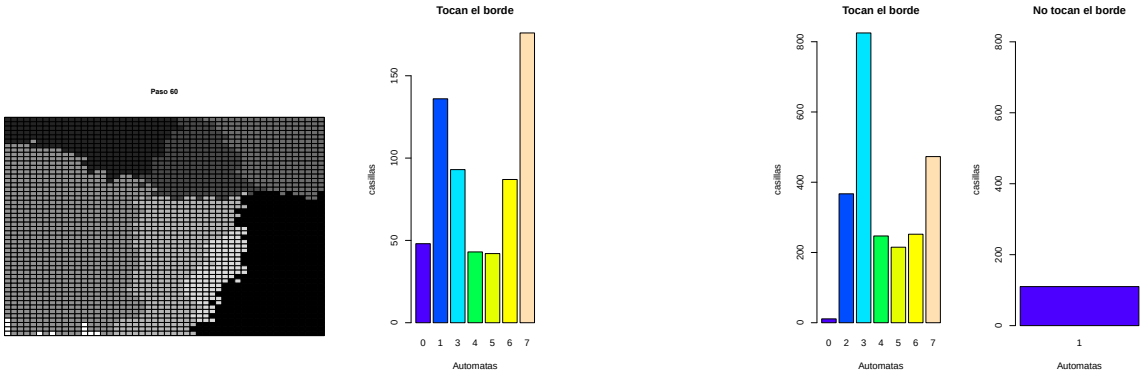
En la figura 3, se presenta las modificaciones realizadas al codigo para este reto.

```
18
19  paso <- function(pos) {
20    fila <- floor((pos - 1) / dim) + 1
21    columna <- ((pos - 1) %% dim) + 1
22    vecindad <-  actual[max(fila - 1, 1) : min(fila + 1, dim),
23                        max(columna - 1, 1): min(columna + 1, dim)]
24     if (actual[fila,columna]==0) {
25      for ( aq1 in vecindad )
26      {
27      if(aq1!=0){
28        if(runif(1)<proba){
29         return(aq1)
30          }
31        }### Primer IF
32      }
33       return(0)
34      }              ### If externo
35   if (actual[fila,columna]!=0){
36     return(actual[fila,columna])
37    }
38  }
39  cluster <- makeCluster(detectCores() - 1)
40  clusterExport(cluster, "dim")
41  clusterExport(cluster, "paso")
42  clusterExport(cluster, "proba")
43
44     actual<- matrix(0,  nrow=dim, ncol=dim)
45     for ( i in 1: length(nucleos)){
46     actual[(round((dim)*runif(1))),(round((dim)*runif(1)))]=nucleos[i]  }
47     salida = paste("p2_t", 0, ".png", sep="")
48     png(salida)
49     plot.sociomatrix(actual, diaglab=FALSE, main="Paso 0",drawlab=F)
50     graphics.off()
51
52  for (iteracion in 1:(dim+floor(dim*0.5)) ) {
53        clusterExport(cluster, "actual")
54        siguiente <- parSapply(cluster, 1:num, paso)
55        actual <- matrix(siguiente, nrow=dim, ncol=dim, byrow=TRUE)
```

Figure 3: Codigo modificado para el reto 1

En la figura 4, se muestra la ultima iteracción para una dimensión de 50, por compatibilidad no se coloco en este reporte el gif generado pero en la carpeta (Practica_2), las diferentes graficas de toda la experimentación. Se ha preferido presentarlas en formato "pdf" por la calidad con que termina la imagen. En la figura 4b contiene los automatas que tocaron el borde para una dimensión de 25*25. En donde todos llegaron a tocar el borde, por lo cual su crecimiento fue detenido por la cuadricula. Ademas la figura 7b, muestra la distribucción de cada uno de los automatas en la cuadricula, segmentando si toca o no el borde de la misma para una dimensión de 50*50.

[2]El codigo fuente de está seccion se ha nombrado "P2_Primer_Reto.R"



(a) 10*10 (b) 25*25 (c) 50*50

Figure 4: Múltiples imágenes

4 Generación aleatoria de los automatas a lo largo del ciclo

En está sección se presenta el analisis realizado para el segundo reto que simula el crecimiento de un fractura, una epidemia entre otros casos pero se le permite a dicho automata,nucleo o fractura generarse a lo largo de la experimentación. En el caso anterior todos los automatas comenzaban desde el inicio, en este caso no es asi. [3].

En la figura 5, se presenta las modificaciones realizadas al codigo para este reto.

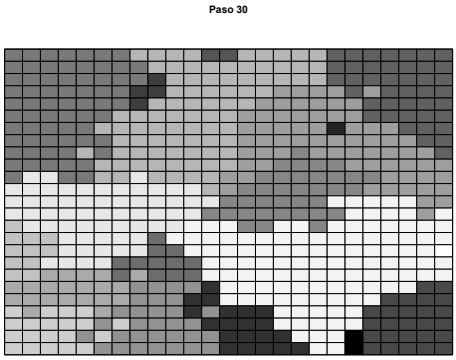
```
cluster <- makeCluster(detectCores() - 1)
clusterExport(cluster, "dim")
clusterExport(cluster, "paso")
clusterExport(cluster, "proba")

nucleos<-seq(1:10)



actual<- matrix(0,  nrow=dim, ncol=dim)
for ( i in 1: length(nucleos)){
  actual[(round((dim)*runif(1))),(round((dim)*runif(1)))]=nucleos[i]  }
  salida = paste("p2_t", 0, ".png", sep="")
  png(salida)
  plot.sociomatrix(actual, diaglab=FALSE, main="Paso 0",drawlab=F)
  graphics.off()



for (iteracion in 1:(dim+floor(dim*0.2))) {
  clusterExport(cluster, "actual")
  siguiente <- parSapply(cluster, 1:num, paso)

  actual <- matrix(siguiente, nrow=dim, ncol=dim, byrow=TRUE)
  if( runif(1)<proba2){
    nucleos<-c(nucleos,length(nucleos)+1)
    fila1<-runif(1)
    colum1<-runif(1)
    if(actual[(1+floor((dim)*fila1)),(1+floor((dim)*colum1))]==0) {
  actual[(1+floor((dim)*fila1)),(1+floor((dim)*colum1))]<-length(nucleos)
  }
  }
  salida = paste("p2_t", iteracion, ".png", sep="")
  tiempo = paste("Paso", iteracion)
  png(salida)
  plot.sociomatrix(actual, diaglab=FALSE, main=tiempo, drawlab=F)
```

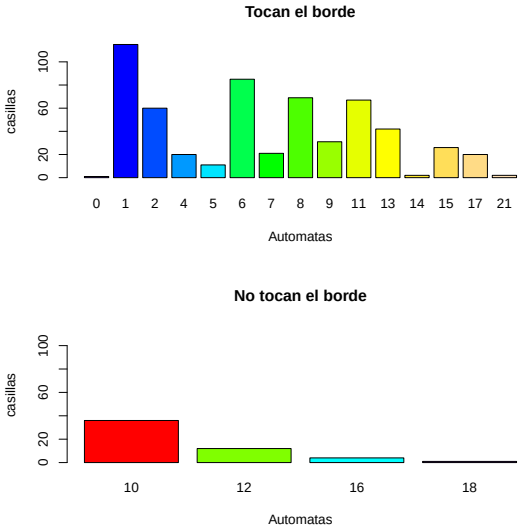
Figure 5: Codigo modificado reto 2

Al principio de la simulación se generarón diez automatas de los cuales el 80 % tocan el borde de la cuadricula, por lo cual su crecimiento es limitado por las condiciones. Como se observo en el caso anterior pocos automatas no tocan el borde, para los dos casos la probablidad de propagación es la misma para poder inferior holisticamente.

En las figuras 6,7, se presentan la distribucción tanto para cuadriculas de 25*25 como 50*50. En donde se puede notar que al aumentar las dimensiones aumentan los automas que no pueden tocar el borde para los casos estudiados en igual proporción.

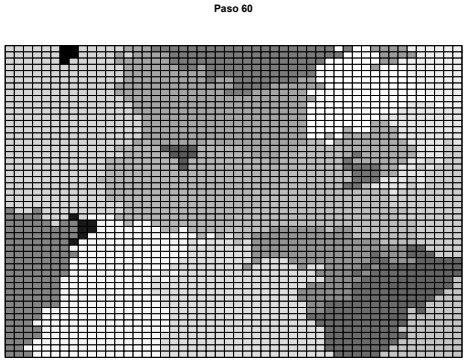
[3]El codigo fuente de está seccion se ha nombrado "P2_Segundo_Reto.R"

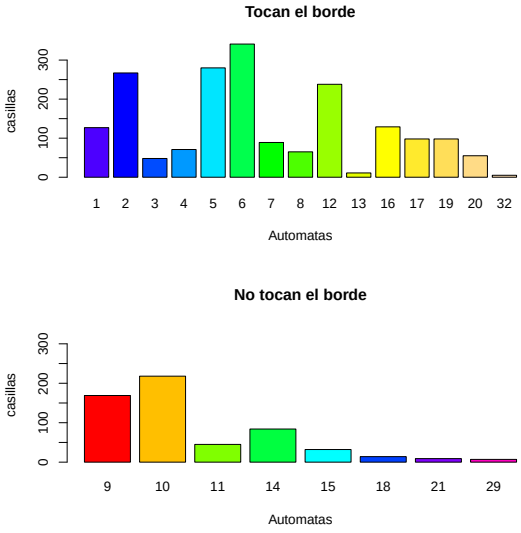




(a) 10*10 (b) 25*25

Figure 6: Múltiples imágenes 25*25





(a) 10*10 (b) 25*25

Figure 7: Múltiples imágenes 50*50