Tarea 2 Autómata celular

Citlali Maryuri Olvera Toscano
Simulación de Sistemas

22 de agosto de 2017

Introducción

Un autómata celular es una colección de células coloreadas en una cuadrícula de forma especifica que evoluciona a través de un número de pasos de tiempo discretos de acuerdo con un conjunto de reglas basadas en los estados de células vecinas. Las reglas se aplican de forma iterativa durante el tiempo que desee.

En la segunda práctica trabajamos con autómatas celulares en dos dimensiones, particularmente el famoso juego de la vida. El estado del autómata se representa con una matríz booleana. Cada celda es o viva (uno) o muerta (cero). En cada paso, la supervivencia de cada celda se determina a partir de los valores de sus vecinos. La regla de supervivencia es sencilla: una celda está viva si exactamente tres vecinos suyos están vivos.

Para esta tarea se nos pide realizar un experimento para determinar el número de iteraciones que dura la simulación sin que se mueran todas las celdas en función de la probabilidad inicial de celda viva.

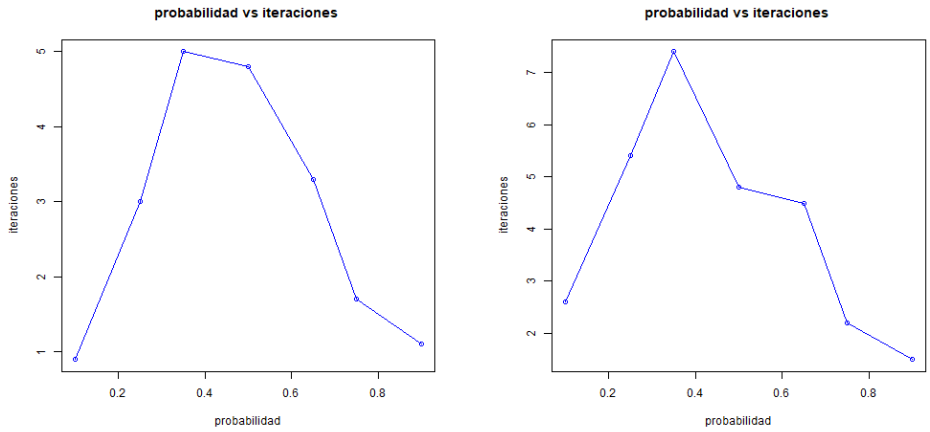
Tarea

La sigueiente tarea se realizó en una máquina con las siguientes especificaciones. Procesador Intel(R)Core(TM) i5-6200U CPU 2.30 GHz 2.40 GHz con 8GB en memoria RAM y sistema operativo Windows 10 Home.

El código realizado para esta tarea es similar al proporcionado por la profesora la única modificación relevante es que ahora podemos determinar la probabilidad de celdas vivas al inicio del experimento. A continuación se muestra esa pequeña parte del código.

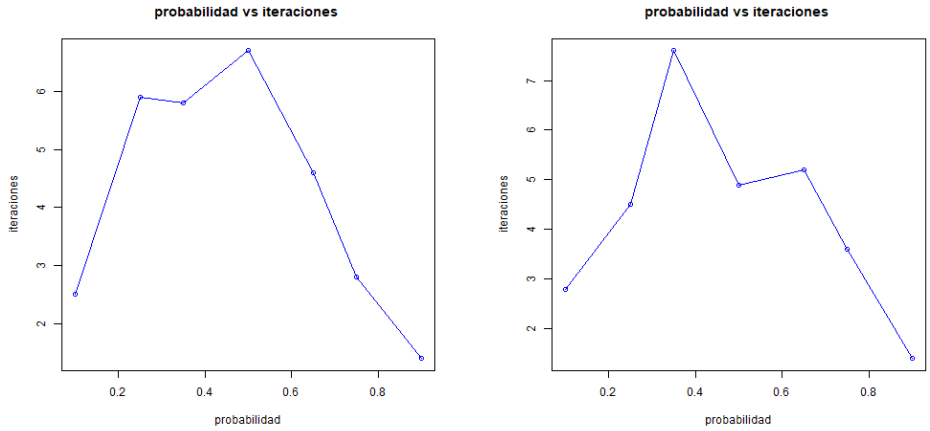
```
proba <- s
actual <- matrix(round(runif(num)<proba)*1, nrow=dim, ncol=dim)
```

Para el experimento consideramos cuatro diferentes dimensiones las cuales son: 10, 20 ,30 y 40, y para las probabilidades consideramos siete valores distintos 0.1, 0.25, 0.35, 0.5, 0.65, 0.75 y 0.9. Cada una de las dimensones con cada una de las probabilidades se repitió diez veces para obtener así un promedio de las iteraciones. Los resultados obtenidos se pueden apreciar en las siguientes gráficas. Además dentro de los archivos de esta tarea se encuentra un GIF animado con nombre *probabilidad 25* el cual muestra los pasos de una simulación con dimensión 10 y probabilidad de celda viva inicial de 0.25.



a) b)

Figura 1: Iteraciones con distintas probabilidades para dimensiones de a): 10 ; b) 20



c) d)

Figura 2: Iteraciones con distintas probabilidades para dimensiones de c): 30 ; d) 40

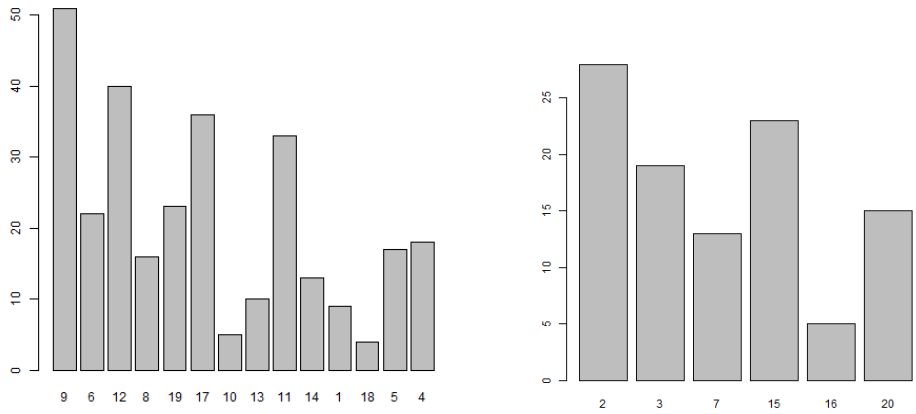
Como podemos observar en las figuras 1 y 2 para las dimensiones de 10, 20 y 40 el mayor número de iteraciones se alcanzó cuando la probabilidad es de 0.35; en cambio con dimensión 30 este máximo valor es alcanzado cuando el valor de la probabilidad es de 0.5. Como era de esperarse se puede verificar que cuando la probabilidad es muy baja o muy alta el número de iteraciones en ambos casos es muy bajo.

Reto 1

El primer reto es modificar la simulación para que modele algún tipo de crecimiento en la microestructura de un material, creando una simulación que establece la forma en que núcleos que inician cada uno en una sola celda van expandiendo a celdas vecinas hasta ocupar el espacio completo, añadiendo todos los núcleos al inicio y expandiendo todos con la misma tasa de crecimiento. Además examinar la distribución de los tamaños de los núcleos que no toquen el borde al finalizar la simulación.

El código sufre varias modificaiones notables ya que desde un inicio se determinan la cantidad de núcleos. Además una vez posicionados en una celda la función *paso* con probabilidad de 0.4 de crecimiento determina hacía que celda vecina expandirse. Ojo: no se invade todos los vecinos con celda vacía si no que dentro de su vecindad elige una celda vacía y con cierta probabilidad la invade.

Ahora bien queremos ver el tamaño de aquellos núcleos que no tocan el borde cuando se finaliza la simulación. Nos tomamos como ejemplo una malla con dimensión 20 y de igual cantidad de núcleos iniciales. En seguida se muestran las gráficas con los resultados.



a) b)

Figura 3: Tamaño de los núcleos que a): tocan el borde ; b)no tocan el borde

Como podemos notar en la Figura 3 los núcleos que no tocan el borde son menos que aquellos que si lo tocan además de que el tamaño de estos núcleos también permanece por debajo en comparación con el tamaño de los núcleos que tocan el borde llegando a ser estos incluso de hasta el doble.

Dentro de los archivos de esta tarea se encuentra un GIF animado con nombre *Reto1* el cual muestra una malla de dimesión 20 con 20 núcleos iniciales y donde se generan más en cada iteración.

Reto 2

El segundo reto es modificar lo anterior a que nuevos núcleos puedan aparecer en momentos distintos, no únicamente al inicio, en cualquier celda que no haya sido previamente ocupado por otro núcleo y examinar cómo este cambio afecta a la distribución de los tamaños.

Para este reto se modificó el código anterior teniendo especial enfasís en el color que sería asignado a los núcleos nacientes. Todas las actualizaciones pueden ser vistas en el código.

Para comparativa con el reto 1 consideramos como ejemplo una malla con dimensión 20 y de igual cantidad de núcleos iniciales. En seguida se muestran las gráficas con los resultados.

En base a lo observado en las gráficas podemos decir que los núcleos que tocan el borde tienen la mayor cantidad de celdas invadidas además la expan-

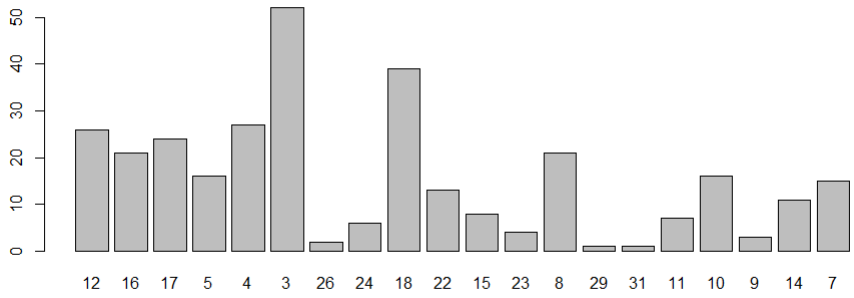


Figura 4: Tamaño de núcleos que tocan el borde

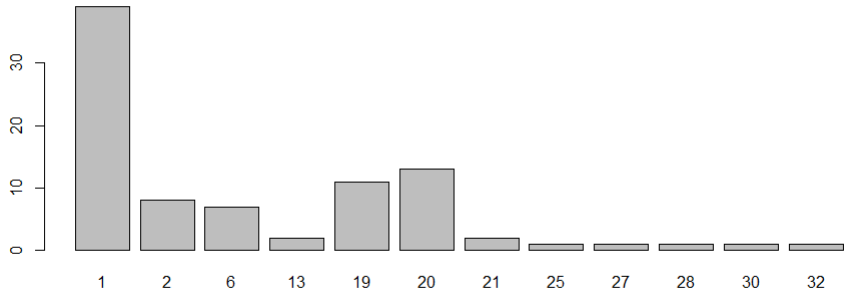


Figura 5: Tamaño de núcleos que no tocan el borde

sión de estos nucleos fue interrumpida por los límites del borde, en cambio los núcleos que no tocan el borde en su mayoría tienen tamaño bastante pequeño esto supongo yo que es porque los núclueos que los rodean llegan a encapsularlos y de esta manera impiden su crecimiento además de que como se van generando nuevos núcleos en cada iteración estos tienen desventaja al poseer menos tiempo de expandirse hasta que se complete la malla.

Dentro de los archivos de esta tarea se encuentra un GIF animado con nombre *Reto2* el cual muestra una malla de dimesión 20 con 20 núcleos iniciales y donde se generan más en cada iteración.