UNAM
Facultad de Ingeniería
Ingeniería Mecatrónica
Tesina de Licenciatura- Titulación por actividad de Investigación (II)

Título**:**Análisis de microarreglos de DNA utilizando redes booleanas (RBN) y redes de probabilidad de Bayes mediante un programa desarrollado en C.

Nombre: José Francisco Revuelta Meza
Director: Biol. Gerardo Coello Coutiño
Año: 2012

ÍNDICE

GLOSARIO

Arabidopsis thaliana.- Es un organismo modelo ampliamente utilizado en estudios genéticos

CSV (comma separated value).- Es un archivo de texto para representar datos en forma de tabla. Las columnas se separan por coma (o por otro valor predeterminado, como tabulador) y los renglones se separan por salto de linea.

Codon.- La secuencia de 3 nucleótidos

Dogma Central de la biología molecular.- Los genes se encuentran en el DNA y se copia la información en mRNA. El mRNA va los ribosomas y por cada codon un aminoácido especifico se agrega a la proteína en formación. Por cada triplete de nucleótidos (codon) se va añadiendo un aminoácido a la proteína siguiendo las normas del código genético

DNA.- Ácido desoxirribonucleico. Molécula de doble cadena en la que se almacena el material genético (genes).

Gen.- Secuencia especifica de DNA que codifica para una proteína en particular.

GEO.- Gene Expression Omnibus. Base de datos con miles de experimentos de microarreglos con acceso libre.

Microarreglo.- Técnica en la cual, físicamente, se imprimen secuencias complementarias del mRNA de un organismo dado. El resultado son los niveles de expresión de las proteínas

mRNA (mensajero) .- Molécula de RNA lineal (no es de doble cadena) en la cual lleva la especificación del gen que se va a transcribir en una proteína

Nivel de Expresión.- Cantidad de proteína medida. Existen mediciones indirectas de la cantidad de proteína a partir de la cantidad de mRNA

Nucleótidos- La unidad básica del DNA. Azúcar, fosfato y base nitrogenada.

Proteína- Secuencia de aminoácidos con una función especifica. Unidad estructural básica de los seres vivos.

OBJETIVOS

OBJETIVO

Elaborar una aplicación que a partir de datos experimentales de micoroarreglos de DNA obtenga ciclos atractores.

Para llegar a los ciclos atractores se usan los resultados de los microarreglos como números booleanos dentro de la teoría de RBN

Se propone una red de probabilidad condicional de Bayes para eliminar el azar en la teoría de RBN y conseguir resultados con utilidad teórica

OBJETIVO ESPECÍFICO

La aplicación debe obtener la red de regulación génica de cualesquiera datos experimentales, dando los archivos normalizados de las muestras y las interacciones entre genes.

INTRODUCCIÓN

Un microarreglo de DNA nos permite medir los niveles de expresión de miles de genes simultáneamente, inclusive genomas completos.

Por nivel de expresión se entiende la cantidad de mRNA (RNA mensajero) como una medida indirecta de la cantidad de proteína. Los métodos de análisis han evolucionado y hoy en día los microarreglos permiten explorar mucho mas allá que simples niveles de expresión.

Considerando bases de datos bioinformáticas disponibles en internet, se puede relacionar las interacciones entre proteínas (calculadas de manera teórica o experimental) y de esta manera generar redes de regulación genética.

En esta investigación los arreglos de genes se comportan como una red booleana. El valor de 0 asignado a un gen significa que no está expresado y de la misma manera el valor de 1 significa que el gen está expresado.

Se deben usar los genes mas representativos, su nivel de expresión génica y las anotaciones de interacciones entre proteínas. Para el desarrollo de la aplicación se usaron 116 muestras de la planta *Arabidopsis thaliana* y 211 de sus genes.

Durante la investigación se elaboraron 3 funciones diferentes de cambio de estado $x_n(t+1)$ y se prueba su convergencia dentro de la RBN.

Para llegar a la función definitiva se calcula la probabilidad condicional (Bayes) de cada gen x_n con respecto a sus vecinos con la ayuda de las muestras. Esto elimina el azar en la elaboración de $x_n(t+1)$ de la teoría de la RBN. El valor de la probabilidad resultante se toma como un valor comparativo para establecer $x_n(t+1)$.

La red se itera conforme la teoría de RBN, con la diferencia de que $x_n(t+1)$ es propuesta a partir de datos experimentales, hasta encontrar ciclos atractores.

RANDOM BOOLEAN NETWORK (RBN)

ESTRUCTURA

Una red booleana consiste de una cantidad n de nodos x_n . Cada nodo x_n es una variable binaria; $x_n \in \{0,1\}$. El arreglo de nodos se define como:

$$X = \{x_{1,} x_{2,} \ldots, x_n\}$$

Para cada nodo x_n se tiene una numero k tal que $k \leq n$, que indica el numero de conexiones que vienen de otros nodos denominados $x_{vi}(n)$ indicando que es un vecino del nodo x_n , $i = (0,1,2,\ldots,k)$.

Este arreglo de datos se va a llamar $K(n)$.

$$K(n) = \{x_{v1}(n), x_{v2}(n), \ldots, x_{vk}(n)\}$$

Se crea una función de manera aleatoria para predecir en el siguiente estado en el tiempo el comportamiento de cada nodo con respecto a los valores de los nodos conectados a el.

$$x_n(t+1) = f(K(n)) \text{ ....(1)}$$

Ejemplo :

Se tiene una red de 16 nodos $(n=16)$. Estando en el tiempo t .

Al analizar solamente las conexiones del nodo $x_1(t)$:

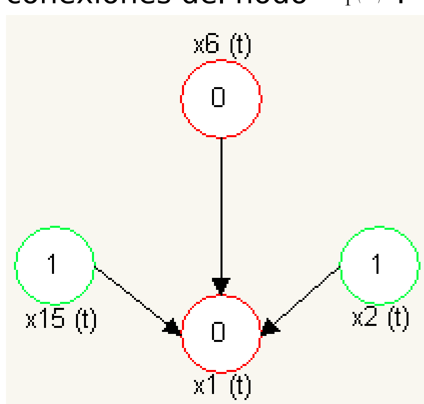


Figura 1. Conexiones de $x_1(t)$

Se tiene que $k=3$; $K(n) = \{x_2 = 1, x_{15} = 1, x_6 = 0\}$ Substituyendo en (1)

$$x_1(t+1) = f(x_2(t) = 1, x_{15}(t) = 1, x_6(t) = 0)$$

Tabla 1. x1 para el estado t+1

x2 (t)	x15 (t)	x6 (t)	f(x2 = 1, x15 = 1, x6 =0)
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

De la tabla, se tiene que:

$$x_1(t+1)=1$$

Los valores de $k, K(n)$ al igual que el comportamiento de la función $x_n(t+1)$, son establecidas como parte de la estructura de la RBN y no se modificarán con el paso del tiempo.

ITERACIÓN

Si se toma en cuenta el arreglo de nodos X con respecto al tiempo, se puede redefinir el arreglo de datos como:

$$X(t)=\{x_1(t),x_2(t),\ldots,x_n(t)\}$$

A cada elemento $x_i(t)\in X(t) \mid i=1,2,\ldots,n$ se le va a aplicar la función $x_n(t+1)$ y cada valor obtenido se va a almacenar en el arreglo $X(t+1)$

$$x_i(t+1)\in X(t+1) \mid X(t+1)=\{x_1(t+1),..,x_n(t+1)\} \quad i=1,2,\ldots,n$$

Se necesita establecer un arreglo $ESTADOS$ con los estados $X(t+i)$, $i=0,1,2,\ldots,r$ para poder analizar los cambios de estados de la RBN en el tiempo.

$$ESTADOS=\{X(t+0),\ldots,X(t+r)\}|i=0,1,2,\ldots,r \quad \text{.....(2)}$$

CICLO ATRACTOR

Los estados $X(t+i)$ se van a comparar entre sí buscando que la red converja en ciclos estables. Estos ciclos de estados se llaman ciclos atractores.

Una vez que el arreglo $ESTADOS$ llega a un ciclo atractor, las iteraciones posteriores no saldrán de este ciclo.

Estos ciclos atractores nos revelan información de los microarreglos que se van a analizar. Se va a tener la referencia de posibles estados de los genes para su interpretación teórica

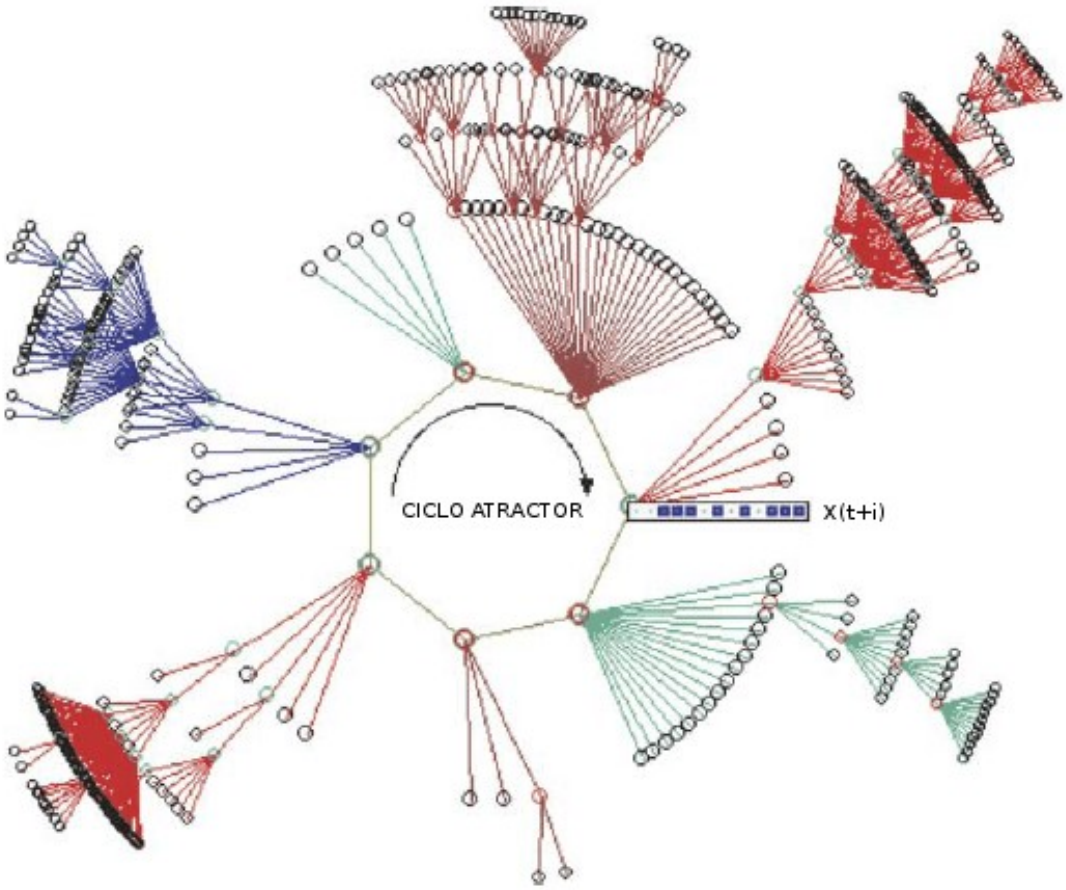


Figura 3. Ciclo atractor. [4]

Se denota cada nodo de la Figura 3 como un estado completo $X(t+i)$. El cuadro blanco indica el valor binario 0 y el cuadro azul indica el valor binario 1 para cada gen de X(t+i).

Cada linea entre nodos es la evolución del nodo en el tiempo hasta llegar al ciclo atractor. En el ejemplo se aprecia que ciclo atractor consta de 7 estados.

Se puede entender que el ciclo atractor es un subconjunto de $ESTADOS$; $CICLO\ ATRACTOR \subset ESTADOS$ -

Para el ejemplo de la *Figura 3* el ciclo atractor se expresa:

$$CICLO\ ATRACTOR = \{X(t+i), X(t+i+1), X(t+i+2), \ldots, X(t+i+6)\}$$

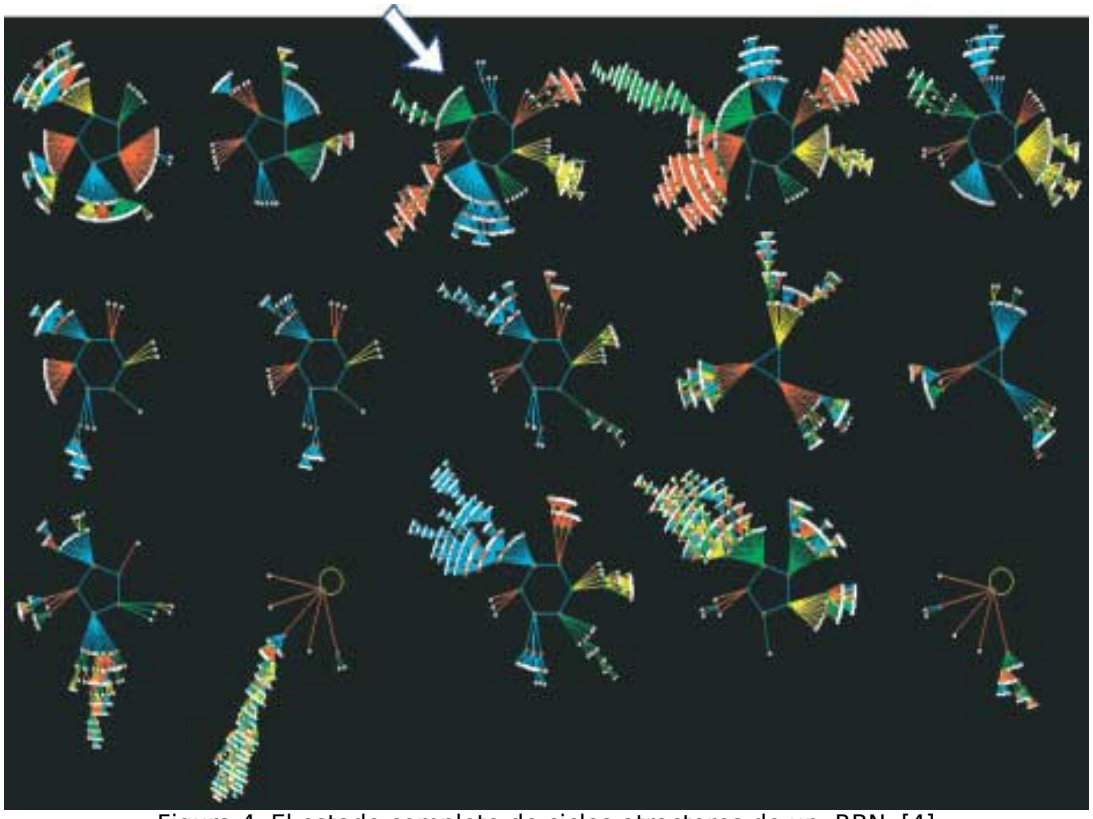


Figura 4 El estado completo de ciclos atractores de un RBN. [4]
La flecha en el esquema muestra el ciclo atractor de la Figura 3.

Para encontrar todos los ciclos atractores de una RBN como en la figura 4, se toma cada posible combinación de los estados binarios de los elementos $x_n(t+i)$ de los arreglos $X(t+i)$ en $i=0$ para comenzar a iterar la red.

Es decir, se tienen 2^n diferentes estados iniciales $X(t+i)|i=0$ para encontrar todos los diferentes ciclos atractores de la red.

Como se ve en la *Figura 4* diferentes estados iniciales $X(t+i)|i=0$ pueden converger en el mismo ciclo atractor.

CONVERGENCIA

La aplicación desarrollada para la investigación tiene un criterio para encontrar el arreglo de datos $CICLOATRACTOR$ dentro del arreglo de estados $ESTADOS$ tal que $CICLOATRACTOR \subset ESTADOS$

Teniendo la ecuación (2):

$$ESTADOS=\{X(t),X(t+1),X(t+2),\ldots,X(t+r)\}$$

Se puede decir que 2 estados $X(t+i)\ y\ \mathrm{X}(t+l)\ \mid\ i\neq l$ son iguales si se cumple que $x_m(t+i)=x_m(t+l)$ para $m=0,1,\ldots,n$

De la ecuación (2), se tiene $ESTADOS=\{X(t+i)\}|i=0,1,2,\ldots,r$.

Se buscan 3 estados iguales $X(t+i)=X(t+l)=(X+p)\ \mid\ i\neq l\neq p$, se compara que los intervalos entre estos estados sean iguales . Sí $i<l<p$ entonces la igualdad se debe cumplir $l-i=p-l$. El numero de estados entre $X(t+i)$ y $X(t+l)$ debe ser igual al numero de estados entre $X(t+l)$ y $X(t+p)$ para encontrar el ciclo atractor.

Una vez que se cumple esta condición, se comparan entre sí todos los estados intermedios entre $X(t+i),X(t+l),(X+q)\ \mid\ i\neq l\neq p$ de la siguiente manera y tomando la definición de igualdad entre estados:

$$X(t+(i+v))=X(t+(l+v))\quad\mid\quad v=0,1,2,\ldots,(l-i)$$

Sí cumple, se tiene:

$$CICLOATRACTOR=\{X(t+i),\ldots,X(t+l)\}$$

Esta arreglo indica la convergencia de nuestra red de RBN. El ciclo atractor es el elemento fundamental del que se derivarán las interpretaciones teóricas.

PROBABILIDAD

PROBABILIDAD CONDICIONAL

Partiendo del concepto básico del cálculo de la probabilidad de la intersección de dos conjuntos A y B en un diagrama de Venn, como se muestra en la Figura 5.

La parte que se va a tomar para la probabilidad condicional es la parte que esta indicada con amarillo.

Ésta es la probabilidad de A y B $P(A \cap B)$.

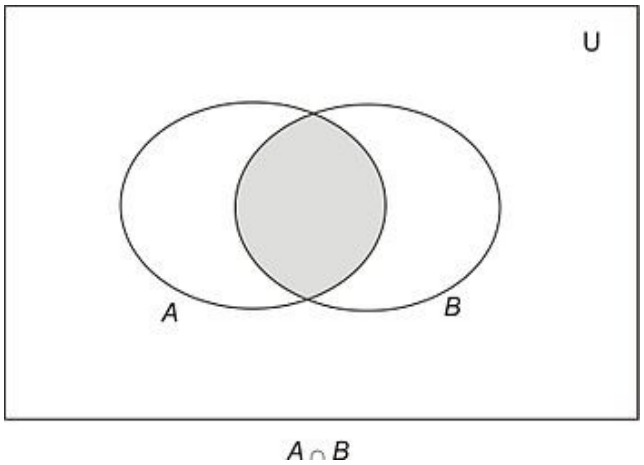


Figura 5. Intersección de conjuntos

El siguiente paso es entender el modelo básico de la probabilidad condicional.

Si se intenta comprender con conjuntos, la probabilidad de A dado B es igual a la probabilidad de A y B entre la probabilidad de B que ya se conoce y ahora es el universo.

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

BAYES

El teorema de Bayes nos ayuda a proponer una probabilidad condicional *a posteriori* P(A|B) de un evento *a priori* P(A) del que se conoce información adicional P(B|A).

En el caso de esta investigación, se va a proponer la probabilidad del estado un gen conociendo el estado de sus vecinos.

Teorema de Bayes:

$$P(A|B)=\frac{P(B|A)\cdot P(A)}{P(B)} \quad \text{..... (3)}$$

Se puede ver que en el primer término de la ecuación (3) se tiene la definición de probabilidad condicional:

Nota:
La nomenclatura se entenderá tomando en cuenta $P(B)+P(\neg B)=1$, el 100%.

$P(B)$ es la probabilidad de B cuando ésta toma el valor de 1 binario
$P(\neg B)$ es la probabilidad de B cuando ésta toma el valor de 0 binario.

Se calcula la probabilidad $P(A\wedge B)$ contando el numero de veces que este evento (A=1 y B=1) se repite en las muestras, dividido entre el numero total de muestras.

$$P(A|B)=\frac{\frac{\#(B=1\wedge A=1)}{\#(Muestras)}}{P(B)} \quad \text{.....(5)}$$

El valor de $P(B)$:

$$P(B)=\frac{\#(B=1)}{\#(Muestras)}$$

IMPLEMENTACIÓN DE BAYES EN LA INVESTIGACIÓN

Substituyendo variables en la ecuación (5)

$$P(A|B)=\frac{\frac{\#(B=1\wedge A=1)}{\#(Muestras)}}{P(B)} \rightarrow P(gen|vecino)=\frac{\frac{\#(vecino=1\wedge gen=1)}{\#(Muestras)}}{P(vecino)} \quad \text{.....(6)}$$

En esta investigación se propuso una red de Bayes de solo un nivel de profundidad tomando los padres del gen que se esté iterando como nodos independientes entre sí.

Este modelo propuesto, conociendo la red de Bayes y sus conexiones, da la mejor manera de explicar los datos observados experimentalmente.
Tomando el mismo ejemplo de la Figura 1:

Se toma el gen x_1 y sus vecinos son los genes $x_{15,}x_6x_2$.

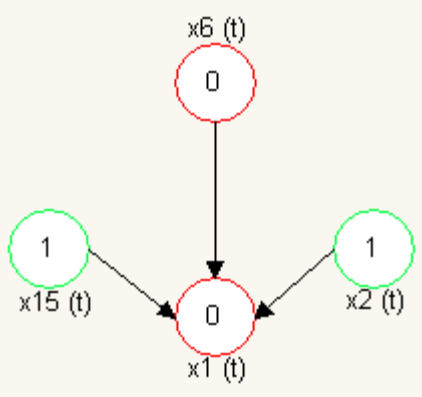


Figura 1: Conexiones de x1

Esta configuración nos dice que la probabilidad del gen x_1 para $(t+1)$

Esta operación se define como:

$$P(x_n)(t+1)=\prod_{i=1}^{i=k} P(x_n|x_{ji}n) \quad \text{.....(7)}$$

Continuando con el ejemplo de la Figura 1, se tiene una tabla con cada gen x_n y su resultado en las diferentes muestras M_i

Tabla 2. Cada M_i es una muestra diferente.

	M_1	M_2	M_3	M_4
x_1	1	1	1	0
x_2	1	0	1	1
x_6	0	1	0	1
x_{15}	1	1	0	0

Nota:
El ejemplo de la tabla tiene el mismo formato que el archivo edo0.csv. En el archivo, cada M_i corresponde al resultado de un microarreglo.

De la ecuación (6):

$$P(gen|vecino)=\frac{\frac{\#(vecino=1\wedge gen=1)}{\#(Muestras)}}{P(vecino)}$$

Tomando gen como x_1 y su vecino x_2 se substituye:

$$P(x_1|x_2)=\frac{\#\frac{(x_2=1\wedge x_1=1)}{(4)}}{\#\frac{(x_2=1)}{(4)}} \rightarrow P(x_1|x_2)=\frac{\#(x_2=1\wedge x_1=1)}{\#(x_2=1)}$$

Contando de la Tabla 2, se tiene que el evento $(x_2=1\wedge x_1=1)$ ocurre 2 veces, en M_1 y M_3 y el evento $(x_2=1)$ ocurre 3 veces, en M_1, M_3 y M_4. Por lo tanto

$$P(x_1|x_2)=\frac{2}{3}=0.666$$

Haciendo el mismo procedimiento para $P(x_1|\neg x_6)$ y $P(x_1|x_{15})$

$$P(x_1|\neg x_6)=\frac{2}{2}=1 \;\; ; \;\; P(x_1|x_{15})=\frac{2}{2}=1$$

Aplicando la ecuación (7)

$$P(x_1)(t+1)=P(x_1|x_{15})\cdot P(x_1|\neg x_6)\cdot P(x_1|x_2)=(1)(1)(0.666)=0.666$$

Se concluye que la probabilidad $P(x_1)(t+1)=0.666$ si sus vecinos $x_2=1, x_6=0, x_{15}=1$.

MODIFICACIÓN DE LA TEORÍA RBN PARA LA INVESTIGACIÓN

Se va a substituir la creación al azar de los valores de $X, n, K(n)$ por datos obtenidos de manera teórica.

Se va a proponer una función determinística $x_n(t+1)$ con un umbral obtenido de una red de probabilidad de Bayes.

El numero de nodos n será el numero de genes en el sistema que se va a analizar. Para el arreglo de nodos $X=\{x_1, x_{2,}..,x_n\}$, cada nodo $x_i|i=0,1,\ldots,n$ es un gen del sistema que se quiere analizar.

Los vecinos $x_{vi}(n)$ para $i=(0,1,2,\ldots,k)$ de cada nodo (gen) x_n , elementos de $K(n)$, van a ser los genes que influyen en la activación de cada gen $x_i|i=0,1,\ldots,n$. Estas conexiones entre genes son obtenidos de manera teórica

IMPLEMENTACIÓN DE ARCHIVOS

Se van a usar 2 archivos: edo0.csv y vecinos.txt

edo0.csv : Es un archivo CSV que contiene el resultado de los microarreglos.

Este archivo esta normalizado para tener como columnas todas muestras y como renglones los genes a analizar.

De este archivo se van a tomar todos los elementos x_n de la muestra deseada y se va a guardar como $X(t+i)|i=0$, el estado inicial para iterar.

vecinos.txt: Es un archivo de texto con los arreglos de datos que establecen las conexiones entre genes $K(n)$ para todo $x_i|i=1,2,..,n$.

Para calcular las ecuaciones (6) y (7) , fundamentales en la implementación de la función $x_n(t+1)$, se usan los 2 archivos.

$$P(x_n|x_{vi}(n))=\frac{\frac{\#(x_{vi}(n)=1\wedge x_n=1)}{\#(muestras)}}{P(x_{vi}(n))} \quad \text{.....(6)}$$

$$P(x_n)(t+1)=\prod_{i=1}^{i=k} P(x_n|x_{vi}\,n) \quad \text{.....(7)}$$

Se tienen los valores de cada vecino $x_{vi}(n)|i=1\ldots k$ para cada gen x_n para poder aplicar la multiplicatoria de la ecuación (7)

Para cada vecino $x_{vi}(n)$ de $x(n)$ se cuentan las veces que $x_{vi}(n)1\wedge x_n=1$ al igual que el cálculo de $P(x_{vi}(n))$ en todas las muestras del archivo edo0.csv.

MANEJO DE ARCHIVOS

Los primeros módulos de la aplicación trabajan con los archivos de la siguiente manera:

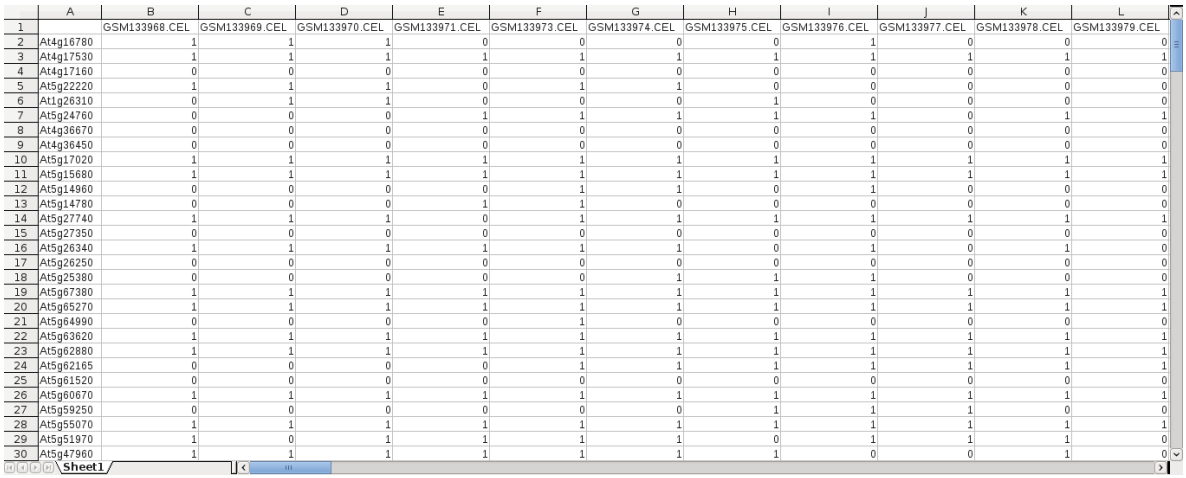
Para el archivo edo0.csv lee columnas y renglones con el valor de separación que en este caso es el tabulador \t y el fin de linea que es \n.

Cada columna es una muestra y cada renglón es un gen.

Para el archivo vecinos.txt se usa cada linea valor de gen, es decir, si se tienen n genes en el sistema, se van a tener a tener n renglones en el archivo vecinos.txt

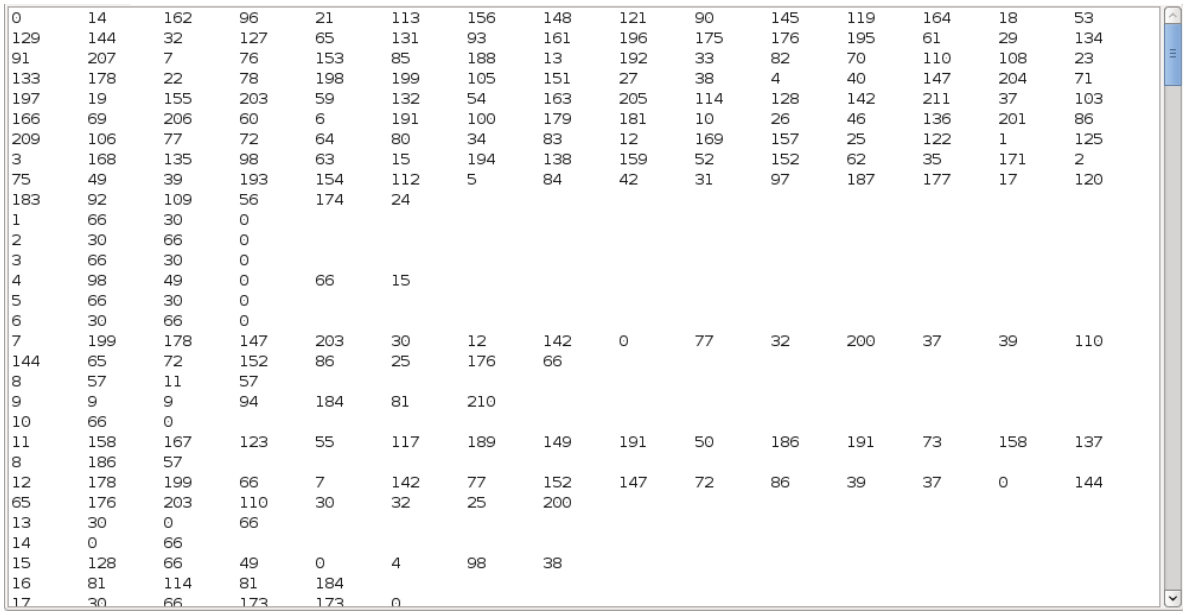
Cada renglón n , que equivale al gen x_n , contiene el arreglo de datos de vecinos K_n

IMÁGENES DE LOS ARCHIVOS



		GSM133968.CEL	GSM133969.CEL	GSM133970.CEL	GSM133971.CEL	GSM133973.CEL	GSM133974.CEL	GSM133975.CEL	GSM133976.CEL	GSM133977.CEL	GSM133978.CEL	GSM133979.CEL
2	At4g16780	1	1	1	0	0	0	0	1	0	0	0
3	At4g17530	1	1	1	1	1	1	1	1	1	1	1
4	At4g17160	0	0	0	0	0	0	0	0	0	0	0
5	At5g22220	1	1	1	0	1	1	0	0	0	0	0
6	At1g26310	0	1	1	0	0	0	1	0	0	0	0
7	At5g24760	0	0	0	1	1	1	1	1	0	1	1
8	At4g36670	0	0	0	0	0	0	0	0	0	0	0
9	At4g36450	0	0	0	0	0	0	0	0	0	0	0
10	At5g17020	1	1	1	1	1	1	1	1	1	1	1
11	At5g15680	1	1	1	1	1	1	1	1	1	1	1
12	At5g14960	0	0	0	0	1	1	0	1	0	0	0
13	At5g14780	0	0	0	1	1	0	0	0	0	0	0
14	At5g27740	1	1	1	0	1	1	1	1	1	1	1
15	At5g27350	0	0	0	0	0	0	0	0	0	0	0
16	At5g26340	1	1	1	1	1	1	0	1	0	1	0
17	At5g26250	0	0	0	0	0	0	0	0	0	0	0
18	At5g25380	0	0	0	0	0	1	1	1	0	0	0
19	At5g67380	1	1	1	1	1	1	1	1	1	1	1
20	At5g65270	1	1	1	1	1	1	1	1	1	1	1
21	At5g64990	0	0	0	0	1	0	0	0	0	0	0
22	At5g63620	1	1	1	1	1	1	1	1	1	1	1
23	At5g62880	1	1	1	1	1	1	1	1	1	1	1
24	At5g62165	0	0	0	0	1	1	1	1	1	1	1
25	At5g61520	0	0	0	0	0	0	0	0	0	0	0
26	At5g60670	1	1	1	1	1	1	1	1	1	1	1
27	At5g59250	0	0	0	0	0	0	1	1	1	0	0
28	At5g55070	1	1	1	1	1	1	1	1	1	1	1
29	At5g51970	1	0	1	1	1	1	0	1	1	1	0
30	At5g47960	1	1	1	1	1	1	1	0	0	1	0

Figura 6. Edo0.csv



```
0 14 162 96 21 113 156 148 121 90 145 119 164 18 53
129 144 32 127 65 131 93 161 196 175 176 195 61 29 134
91 207 7 76 153 85 188 13 192 33 82 70 110 108 23
133 178 22 78 198 199 105 151 27 38 4 40 147 204 71
197 19 155 203 59 132 54 163 205 114 128 142 211 37 103
166 69 206 60 6 191 100 179 181 10 26 46 136 201 86
209 106 77 72 64 80 34 83 12 169 157 25 122 1 125
3 168 135 98 63 15 194 138 159 52 152 62 35 171 2
75 49 39 193 154 112 5 84 42 31 97 187 177 17 120
183 92 109 56 174 24
1 66 30 0
2 30 66 0
3 66 30 0
4 98 49 0 66 15
5 66 30 0
6 30 66 0
7 199 178 147 203 30 12 142 0 77 32 200 37 39 110
144 65 72 152 86 25 176 66
8 57 11 57
9 9 9 94 184 81 210
10 66 0
11 158 167 123 55 117 189 149 191 50 186 191 73 158 137
8 186 57
12 178 199 66 7 142 77 152 147 72 86 39 37 0 144
65 176 203 110 30 32 25 200
13 30 0 66
14 0 66
15 128 66 49 0 4 98 38
16 81 114 81 184
17 30 66 173 173 0
```

Figura 7. Vecinos.txt

FUNCIÓN_ $x_n(t+1)$

La función para el cambio de estado de cada gen $x_n(t+1)$ se planteó de 3 maneras distintas a lo largo de la investigación.

Para las 3 funciones que se van a definir a continuación se debe tomar en cuenta que todo valor de $x_i(t+1)$ que se vaya a obtener, es parte de su correspondiente arreglo de datos $X(t+1)$,

$$x_i(t+1) \in X(t+1) \quad | \quad X(t+1) = \{x_1(t+1), .., x_n(t+1)\} \quad i = 1,2,\ldots,n$$

que a su vez pertenece al arreglo de datos de ESTADOS

$$X(t+i) \in ESTADOS \quad | \quad ESTADOS = \{X(t+0), .., X(t+n)\} \quad i = 0,2,\ldots,n$$

- La primer manera fue planteada de manera determinística con una sumatoria del estado de sus vecinos.

$$x_n(t+1) = 1 \quad \text{sí} \quad \sum_{i=0}^{i=k} x_{ji} n(t) > \frac{k}{2}$$

$$x_n(t+1) = 0 \quad \text{sí} \quad \sum_{i=0}^{i=k} x_{ji} n(t) < \frac{k}{2}$$

Es decir, cuando en la mayoría de los vecinos de x_n tienen valor 1, el resultado es $x_n(t+1) = 1$ y cuando en la mayoría de los vecinos de x_n tienen valor 0, el resultado $x_n(t+1) = 0$.

Este planteamiento se parece al propuesto en [4] para probar la convergencia de un RBN.

- La segunda manera de definir la función $x_n(t+1)$ fue de manera no determinística con una simulación probabilística

$$P(x_n)(t+1)=\prod_{i=1}^{i=k} P(x_n|x_{vi}n)$$

Para la simulación, según el planteamiento de las ecuaciones (6) y (7)

$$P(gen|vecino)=\frac{\frac{\#(vecino=1\wedge gen=1)}{\#(muestras)}}{P(vecino)}$$

$$P(x_n)(t+1)=\prod_{i=1}^{i=k} P(x_n|x_{vi}n)$$

Del anexo de código, función RBN() pagina 32

Pseudocódigo:

i=0

while i < n (numero de genes)
muestra = 0

while muestra < 116 (numero total de muestras)
if x_i = 1 AND vecino = 1
intersección = intersección + 1
end if

numerador = numerador * (interseccion / muestra)
denominador = denominador * P(vecino)
$P(x_i)$ = $P(x_i)$ * (numerador / denominador)
muestra = muestra + 1

if simulación probabilistica < $P(x_i)$
$x_i(t+1)$ = 1
else
$x_i(t+i)$ = 0
i=i+1

- El planteamiento de la tercer función $x_n(t+1)$ usa el mismo cálculo de la probabilidad de Bayes pero **SIN** la simulación probabilística . El valor de la probabilidad $P(x_n)(t+1)$ se toma como un umbral *y* la función nuevamente se comporta de manera determinística.

Pseudocódigo:

i=0

while i < n (numero de genes)
muestra = 0

while muestra < 116 (numero total de muestras)
if x_i = 1 AND vecino = 1
intersección = intersección + 1
end if

numerador = numerador * (interseccion / muestra)
denominador = denominador * P(vecino)
$P(x_i)$ = $P(x_i)$ * (numerador / denominador)
muestra = muestra + 1

if $P(x_i)$ > umbral
$x_i(t+1)$ = 1
else
$x_i(t+i)$ = 0
i=i+1

RESULTADOS FUNCIÓN $x_n(t+1)$

- RESULTADO FUNCIÓN 1 - Contando

Con el criterio de convergencia, esta función hace que la red converja pero no tiene utilidad teórica.

- RESULTADO FUNCIÓN 2 – Simulación probabilística

Según el criterio de convergencia propuesto, el ciclo atractor no se encuentra, llegando a numero de estados tomando el arreglo

$$ESTADOS=\left\{X\left(t+i\right)\right\}|i=1,2,\ldots,r$$

con valores de r hasta de 2 millones.

CONVERGENCIA

Cuando se hicieron las pruebas con la segunda función $x_n(t+1)$ que usa una simulación probabilística, el programa no converge. Se le hizo una pequeña modificación al criterio de convergencia.

Como se encuentra definida la función en el reporte, todos los elementos de todos los estados de un ciclo deben coincidir con el del siguiente ciclo para concluir que se está en un *ciclo atractor.*

Este criterio se modifico para que en lugar de que TODOS los elementos de todos los estados de un ciclo deban coincidir con el del siguiente ciclo, ahora deben coincidir solo un porcentaje de los elementos para tomar un estado y otro como “iguales” y encontrar el ciclo atractor.

Este valor se probó con valores de $80\%-100\%$.

En el intervalo de $80\%-85\%$ la red converge con muy pocas iteraciones pero el error es muy grande. Si se habla de 211 genes con este intervalo se tiene una diferencia de 20-30 genes entre estado y estado para decir que son “iguales”. La diferencia es muy grande y no es aceptable.

Con el intervalo de $85\%-90\%$ la red puede converger o no según el estado $X(t+i)|i=0$ que se seleccione para comenzar a iterar.

Con valores mayores al 90% la red no converge.

- RESULTADO FUNCIÓN 3 – Usando la probabilidad como umbral

Se elimina la simulación probabilístia del proceso. La función se comporta de manera determinística y al usar el valor de la probabilidad como umbral se tiene una red que converge y una referencia teórica de todas las muestras con las que se trabajan. El valor del umbral se puede variar.

Usando este criterio, se regresa al valor de convergencia del 100% de los elementos entre estado y estado para que sean considerados iguales.

EJEMPLO

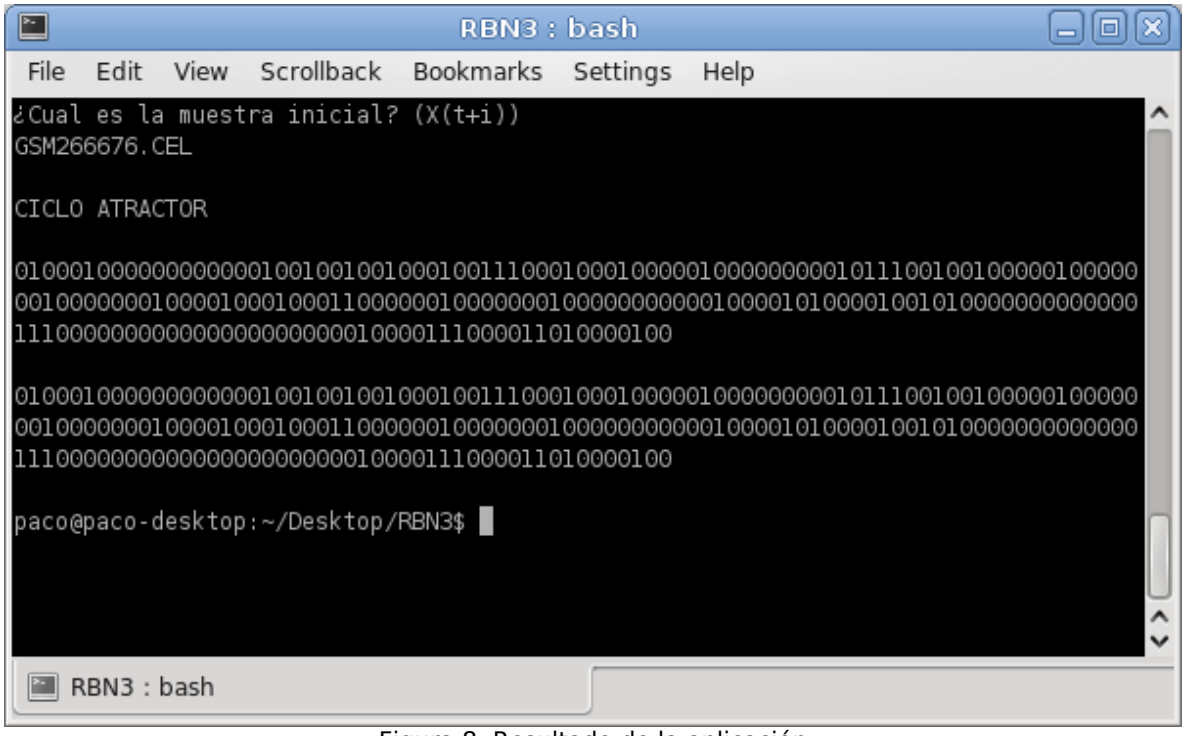


Figura 8: Resultado de la aplicación

1. Se toma la muestra GSM266676.CEL de el archivo *edo0.csv* como estado inicial $X(t+i)|i=0$.

2. Para todo gen $x_i(t) \in X(t) \mid i=1,2,\ldots,n$ se calcula la probabilidad condicional $P(x_n)(t+1)=\prod_{i=1}^{i=k} P(x_n|x_{vi}\,n)$ que se usa como umbral en la función $x_n(t+1)$ y se agrega el estado $X(t+1)$ a $ESTADOS=\{X(t+0),X(t+1)\}$

3. Se aplica el criterio de *CONVERGENCIA* a *ESTADOS.* Si el criterio cumple, el programa imprime el ciclo atractor, si no, se regresa al paso 2 teniendo ahora $X(t+i)|i=i+1$.

RESULTADOS DE LA RED

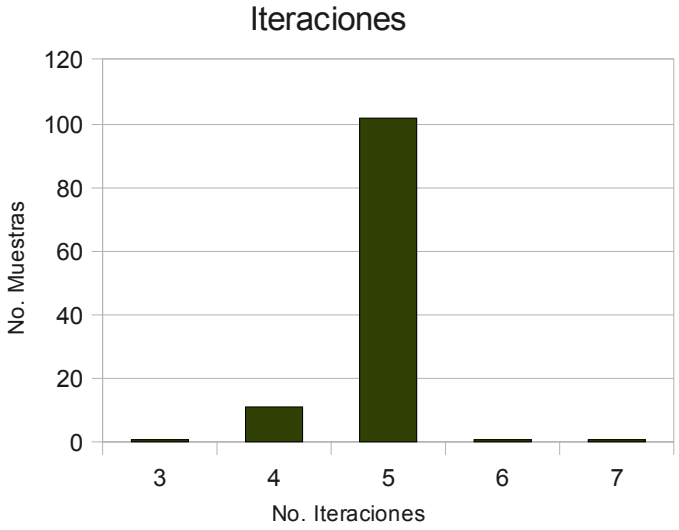
Se usó cada una de las 116 muestras como estado inicial $X(t+i)|i=0.$

Cada muestra inicial obtiene de resultado un ciclo atractor.

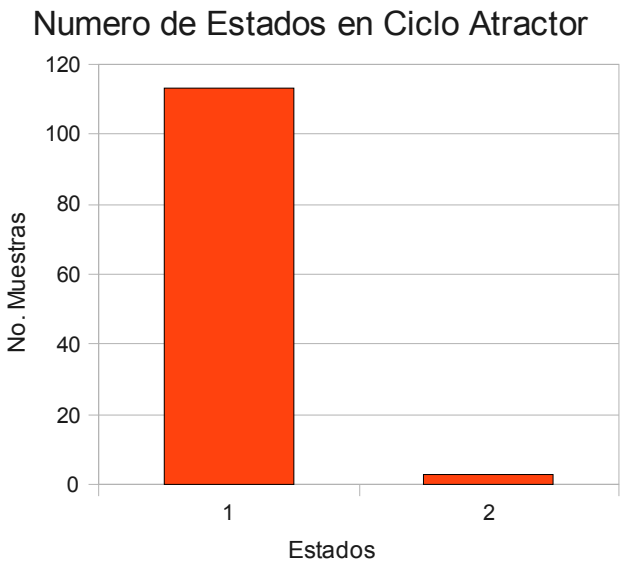
Se tiene que el 88% de las muestras llegaron a un ciclo atractor en 5 iteraciones, el 97.5% de los ciclos atractores constan de 1 estado y se obtuvieron en total 15 ciclos atractores. El 55% de las muestras convergen al mismo ciclo atractor.

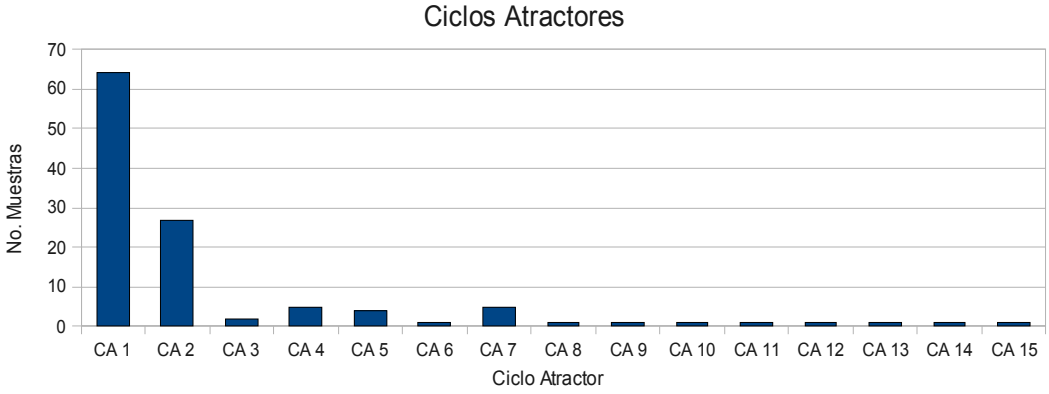
Se muestran los resultados completos:

No. Iteraciones	No. Muestras
3	1
4	11
5	102
6	1
7	1



No. Estados	No. Muestras
1	113
2	3





Ciclo Atractor	CA 1	CA 2	CA 3	CA 4	CA 5	CA 6	CA 7	CA 8	CA 9	CA 10	CA 11	CA 12	CA 13	CA 14	CA 15
No de Muestras	64	27	2	5	4	1	5	1	1	1	1	1	1	1	1

CONCLUSIONES

Se encontró que usando el resultado de la probabilidad de una red de Bayes de un nivel de profundidad, como un valor de comparación en la función $x_n(t+1)$ de una RBN y usando el valor de convergencia definido en la tesina, la aplicación converge y encuentra un ciclo atractor para cada valor inicial $X(t+i)|i=0$ de las 116 muestras que se le de para comenzar a iterar.

El número de iteraciones de cada muestra para llegar al ciclo atractor y el numero de estados en los ciclos atractores fueron muy consistentes.

Esto se debe a que la probabilidad condicional es una multiplicatoria. Si hay alguna probabilidad 0, todo el factor se vuelve 0 ocasionando la rápida convergencia.

Este inconveniente se puede solucionar depurando el archivo de vecinos.txt, definiendo un manejo de la probabilidad por cero y/o modificando la manera de calcular la probabilidad, como se expone en “Trabajo a Futuro”

Aunque se obtuvieron 15 ciclos atractores, hay que tomar en cuenta que el 78% de las muestras convergieron en CA1 y CA2, dando pie a tener una base para comenzar a analizar los resultados de manera teórica.

La aplicación puede trabajar con cualquier resultado de cualquier microarreglo mientras los archivos edo0.csv y vecinos.txt cumplan con la norma establecida.

TRABAJO A FUTURO

La red de probabilidad se propuso con la suposición que los nodos son independientes entre sí.

Este criterio, aunque es simple en su planteamiento, deja fuera las interacciones que pueden tener entre sí los vecinos del gen a analizar.

Como trabajo a futuro propongo modificar el planteamiento de el cálculo de la probabilidad condicional, tomando en cuenta la combinación de los estados de los vecinos y no solamente la probabilidad condicional del gen con un vecino a la vez.

Esta aplicación será utilizada en el Instituto de Fisiología Celular para obtener hacer un análisis teórico de todos los ciclos atractores de muestras de pacientes sanos y enfermos para comparar diferencias y afinidades.

ANEXO-CÓDIGO

```
#include<stdio.h>
#include<string.h>
#include<stddef.h>
#include<stdlib.h>
#include <math.h>

void rbn(int vecinos2[][213],int genes,int estadocero[],long double
probasigen[],int arrarchivo[][116], int columna);

void imprimir(int arrestado[][3000],int genes,int f, int g, int p, int q,int tresmil);

main()

{
  int genes=211;
  int estadocero[212];
  int vecinos[1000][213];
  int vecinos2[1000][213];
  int contador=0,contador2=0,contador3=0;
  char linea[2050];
  FILE*file;
  int columna=0;
  int x=0,y=0,x2=0,x3=0,xvecinos=0,yvecinos=0,resultado=0;
  char experimento[25];
  char numerotemporal[4];
  char rvecinos[2000];
  char nombregen[10];


////////////////////////////////MANEJO edo0.csv
   file=fopen("/home/paco/Desktop/RBN3/edo0","r");
```

```
printf("\n");
printf("¿Cual es la muestra inicial? (X(t+i))\n");
scanf("%s",experimento);
char lineadegenes[2250];
int row=0,col=0,contaldg=0;
int arrarchivo[213][116];

while(fgets(linea,2050,file)!=NULL)
 {
  if(linea[0]=='\t')
     columna=obtencolexp(linea,experimento);
  else
     {
       estadocero[contador]=obtenedo0(linea,columna);
      contador++;
     }
  while(linea[x]!='\t')
     {
      lineadegenes[contaldg]=linea[x];
      contaldg++;
      x++;
     }
  if(row>0)
     {
      lineadegenes[contaldg]=linea[x];
      contaldg++;
     }
  while(col<116&&row>0)
     {
      if(linea[x]=='0' || linea[x]=='1')
        {
          if(row>0)
              {
```

```
                arrarchivo[row-1][col]=linea[x]-48;
                  col++;
                }
            }
           x++;
          }
      row++;
      col=0;
      x=0;
    }
  fclose(file);
  int i=0,j=0;
////////////////////////////// MANEJO vecinos.txt

 file=fopen("/home/paco/Desktop/RBN3/vecinos","r");

   while(fgets(rvecinos,2000,file)!=NULL)
    {
      xvecinos=0;
      for(x=0;x<=2000;x++)
          {
           if(rvecinos[x]=='\t' && rvecinos[x+1]=='\n')
             break;
           if(rvecinos[x]=='\t')
             {
               x2=x+1;
               while(rvecinos[x2]!='\t')
                  x2++;
               for(x3=x;x3<=x2;x3++)
                  {
                    if(rvecinos[x3]!='\t')
                      {
                        numerotemporal[x3-x-1]=rvecinos[x3];
```

```
                    }
                  else
                    numerotemporal[x3-x-1]='\0';
                 }
              resultado=atoi(numerotemporal);
              vecinos[xvecinos][yvecinos]=resultado;
              xvecinos++;
            }
         }
     vecinos[xvecinos][yvecinos]=-1;
     yvecinos++;
   }
 fclose(file);
 contador=0;
 while(contador2<=211)
  {
    contador=0;
    contador3=0;
    while(vecinos[contador][contador2]!=-1)
       contador++;
    x=0;
    for(x=contador;x>=0;x--)
       {
        for(x2=x-1;x2>=0;x2--)
          {
            if(vecinos[x][contador2]==vecinos[x2][contador2])
               break;
            if(x2==0)
               break;
          }
         if(x2==0&&vecinos[x][contador2]!=-1&&vecinos[x][contador2]!
=vecinos[0][contador2])
          {
```

```
              vecinos2[contador3][contador2]=vecinos[x][contador2];
              contador3++;
             }
           if(x==0)
             {
              vecinos2[contador3][contador2]=vecinos[0][contador2];
              vecinos2[contador3+1][contador2]=-1;
             }
          }
      contador2++;
    }
long double probconta=0;
  contador2=0;
  i=0;
  j=0;
 long double probasigen[211];
  for( i=0;i<=211;i++)
    {
      probconta=0;
      for(j=0;j<116;j++)
          {
           if(arrarchivo[i][j]==1)
          probconta++;
          }
      probasigen[i]=probconta/116;
    }
  i=0;
////////////////// SE LLAMA LA FUNCION RBN
  rbn(vecinos2,211,estadocero,probasigen,arrarchivo, columna);
}


/////////////// ACABA EL MAIN
```

```
int obtenedo0(char linea[],int columna)
{
  int contador2=0,contalinea=0,x=0;
  int resultado=0;
  while(contador2!=columna)
    {
      if(linea[x]=='\t')
          contador2++;
      x++;
    }
  if (linea[x]=='1')
    resultado=1;
  else
    resultado=0;
  contalinea++;
  return resultado;
}


int obtencolexp(char linea[],char experimento[])
{
  int columna=0;
  ptrdiff_t numero=0;//cuando se usa la resta con el valor de un apuntador, se
usa ptrdiff_t como tipo de dato
  int x;
  char*substring;
  substring=strstr(linea,experimento);
  numero=substring-&linea[0];//se restan las posiciones en el arreglo de linea
para saber donde esta el experimento
  for(x=0;x<=numero;x++)
    {
      if(linea[x]=='\t')
          columna ++;
    }
  return columna;
```

```
}
//////////////PROCESO RBN
void rbn(int vecinos2[][213],int genes,int estadocero[],long double
probasigen[],int arrarchivo[][116], int columna)


{
  printf("\n");
  int contador=0;
  int contador2=0;
  int y=1,x=0;
  int z=0,w=0,p=0,q=0,g=0,f=0,l=0,m=0,i=0,k=0,e=0,j=0;
  int arrestado[213][3000];
  int renglonestado[212];
  long double numerador=1, denominador=1;
  int muestra=0;
 int exp=116;
 long double conta1=0, conta2=0,proba=0;
int tresmil=0;
int conta3=0;
int estadorepetir[212];
int convergencia=212;
///CONVERGENCIA 100%
int estadosmedios=0;
float temporal=0;
int temporal2=0;


  do
    {
        if(y==3000)
         {
                 for(x=0;x<=genes;x++)
                   arrestado[x][0]=arrestado[x][y-2];
                y=1;
```

```
            tresmil++;
          z=0;w=0;p=0;q=0;g=0;f=0;l=0;m=0;i=0;k=0;e=0;j=0;
        }

    for(x=0;x<genes;x++)/
        {
         z=0;
         contador=0;
         contador2=0;
         numerador=1;
         denominador=1;

////////////////////////////////////SE USA PROBABILIDAD CONDICIONAL
         while(vecinos2[z][x]!=-1)
          {
              for(muestra=0;muestra<exp;muestra++)
              {
              if(arrarchivo[x][muestra]==1 && arrarchivo[vecinos2[z][x]]
              [muestra]==arrestado[vecinos2[z][x]][y-1])
              conta1++;
              }
           proba=conta1/exp;
            numerador*=proba;
            if(arrestado[vecinos2[z][x]][y-1]==0 )
              denominador*=(1-probasigen[vecinos2[z][x]]);
            else if(probasigen[vecinos2[z][x]]>0)
              denominador*=probasigen[vecinos2[z][x]];
           conta1=0;
           proba=0;
           z++;
          }
        temporal=(numerador/denominador)*1000;
        temporal2=(int)temporal;
```

```
                if(temporal2>500)
                {
                arrestado[x][y]=1;
                }
                else

                {
                arrestado[x][y]=0;
                }
        }  //el de xnovi
w=y;
    if(l<=convergencia)
        {
         do
           {
            w--;
            if(w==-1)
                {
                  break;
                }
            l=0;
            conta3=0;
            do
                {
                if(arrestado[conta3][y]==arrestado[conta3][w])//11conta3
                l++;
                conta3++;
                }while(conta3<genes+1);
                if(l>=convergencia)//if(l==genes-1)
                {
                  for(k=0;k<genes;k++)
                    renglonestado[k]=arrestado[k][w];
                }
```

```
        f=w;
        g=y;
   }while(l<=convergencia);
 }//el de if l!=genes-1
  k=0;
if(l>=convergencia&&p==0)//////////////////////11conta3//////////14conta3
 {
  m=0;
  conta3=0;
  do
    {
    f(arrestado[conta3][y]==renglonestado[conta3])
    m++;
   conta3++;
    }while(conta3<genes+1);
  if(m>=convergencia)
    {
        p=y;
    }
}

 conta3 =0;
 if(p!=0&&p!=y)
 {
  i=0;
  do
    {
        if (arrestado[conta3][y]==renglonestado[conta3])
        i++;
        conta3++;
     }while(conta3<genes+1);
        if(i>=convergencia)
    {
```

```
        q=y;}
    }

if(p!=0&&q!=0&&(q-p)==(g-f))
    {
      e=p;
      z=f;
      do
        {
            conta3=0;
         k=0;
         do
            {
            if(arrestado[conta3][z]==arrestado[conta3][e])
            k++;
            conta3++;
            }while(conta3<genes+1);
          if(k>=convergencia)
            estadosmedios++;
          z++;
          e++;
         }while(e<q+1);

            if(estadosmedios<(g-f)-2)
            estadosmedios=0;
    }

    if(p!=0&&q!=0&&(q-p)!=(g-f))
     {
     p=0;
      q=0;
      l=0;
    estadosmedios=0;
```

```
        }
        if((y-p)>(g-f))
        {
        p=0;q=0;l=0;estadosmedios=0;
        }
if(y<100)
{
        p=0;q=0;l=0;estadosmedios=0;
}

 y++;
}while(estadosmedios==0); //////////////CONVERGENCIA
////////// IMPRIMIR RESULTADOS
 imprimir(arrestado,genes,f,g,p,q,tresmil);

}

void imprimir(int arrestado[][3000],int genes,int f, int g, int p, int q,int tresmil)
{
 int j=0,i=0;
  printf("CICLO ATRACTOR");
  if(g==p)
  {
   for(j=f;j<g+1;j++)
        {
          printf("\n\n");
          for(i=0;i<=genes;i++)
            printf("%d",arrestado[i][j]);
        }
  }
printf("\n\n");
}
```

REFERENCIAS

[1] Gershenson, C. 2002. Classification of Random Boolean Networks . *Artificial Life VIII, Standish, Abbass, Bedau (eds)(MIT Press) pp 1–8*

[2] Gershenson, C. 2004. Introduction to Random Boolean Networks

[3] Hawick, K.A., James, H.A, Scogings, C.J. 2007 . Simulating Large Random Boolean Networks.

[4] Bornholdt, S. 2008. Boolean network models of cellular regulation: prospects and limitations. *J. R. Soc. Interface 2008 5, S85-S94*

[5] Aldana, M, Coppersmith, S, Kadanoff L. Boolean Dynamics with Random Couplings

[6] Aldana, M. 2003. Boolean dynamics of networks with scale-free topology. *Physica D 185 (2003) 45–66*