Parcial 3, parte 2: autómata celular unidimensional

2 o 3 participantes por grupo.

Mandar el códigos C a guillaumh@gmail.com, hasta el martes 13/6, 23hs59. Indicar los nombres de los integrantes del grupo.

Resumen: reimplementar el programa dado en última página de este enunciado.

Automata celular unidimensional

Un automata celular unidimensional (ACU) consiste en un arreglo de una dimensión, cuyos elementos (llamados células) pueden estar en dos estados posibles (0 o 1), y cuyo estado general evoluciona en el tiempo.

Esta evolución se hace en tiempo discreto y sincronizado, es decir, en cada paso, todas las células cambian de estado al mismo tiempo.

En un momento t, el estado de las células determina completamente el estado de las células en el momento $t + 1$. Más precisamente, para saber el estado de una célula en posición i del arreglo en un momento $t + 1$, solo se necesita mirar los estados de 3 células en el momento t: las que están en posición $i - 1$, i e $i + 1$.

En este marco, una célula depende solo de sus vecinas inmediatas y de ella misma en el paso anterior. Hay $2^3 = 8$ patrones de 3 células binarias. Para cada patrón, está definido el estado nuevo de la célula afectada en el paso siguiente:

Patrón	111	110	101	100	011	010	001	000
Estado nuevo	?	?	?	?	?	?	?	?

Para definir el comportamiento de un ACU, tenemos que definir la función que nos da esta transición. Es decir, tenemos que fijar los 8 bits de la tabla. Por lo cual, tenemos $2^8 = 256$ funciones distintas (o reglas) de ACU.

En este proyecto vamos a implementar el ACU llamado “Regla 30”.

El nombre de las reglas

Los ACU tienen como nombre la palabra “Regla” seguida de la representación decimal del byte de la segunda línea de la tabla vista más arriba. Es decir, el nombre de un ACU define completamente su comportamiento.

Por ejemplo, el automata definido por:

Patrón	111	110	101	100	011	010	001	000
Estado nuevo	0	0	0	0	0	0	0	0

se llama “Regla 0”, dado que $0_d = 0000000_b$.

El siguiente:

Patrón	111	110	101	100	011	010	001	000
Estado nuevo	0	0	0	0	0	0	0	1

se llama “Regla 1”, dado que $1_d = 0000001_b$.

El siguiente:

Patrón	111	110	101	100	011	010	001	000
Estado nuevo	1	1	1	1	1	1	1	1

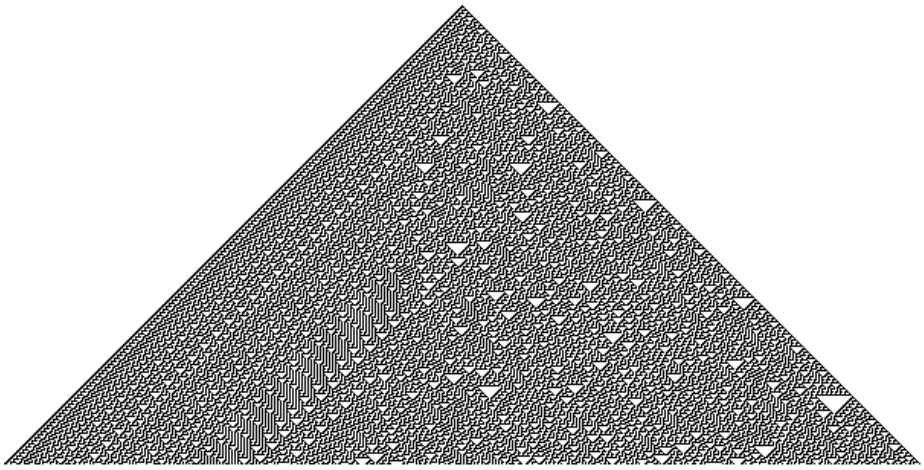
es "Regla 255", dado que $255_d = 11111111_b$.

El automata que nos interesa en este proyecto es Regla 30:

Patrón	111	110	101	100	011	010	001	000
Estado nuevo	0	0	0	1	1	1	1	0

Todos los ACU son totalmente deterministas. Varios de ellos tienen una ejecución simple y predecible. Sin embargo, algunos tienen un comportamiento aparentemente caótico, y Regla 30 es uno de ellos (Regla 110 es otro ejemplo interesante).

Cuando el autómata Regla 30 empieza con una sola célula en estado 1 y el resto a el principio de su *grafo temporal* (es decir la representación gráfica de sus estados línea por línea, desde el tiempo $t = 0$) es el siguiente:



Programa de ejemplo

En la última página de este enunciado se encuentra un ejecutable de Regla 30. Pegar el texto de la última página en el clipboard de JSLinux y luego hacer:

```
uudecode < /dev/clipboard
```

Luego se puede ejecutar `./rule`. La ejecución del programa es infinita, pero se puede interrumpir con `CTRL+c`.

Implementación

El objetivo de este proyecto es implementar un programa que muestre el grafo del automata Regla 30, tal como lo hace el ejecutable proveido en la última página de este enunciado.

El arreglo

Si bien el modelo matemático de un ACU es definido sobre una cinta infinita en las dos direcciones, solo podemos representar una cantidad finita de células en un programa C. La primera y la última células son consideradas como adyacentes. Es decir, en nuestra implementación el ACU va a ser un toro (o cinta circular).

Más concretamente, en el arreglo `c` de tamaño T, el elemento `c[0]` tiene como vecinos los elementos `c[T-1]` y `c[1]`. Por otro lado el elemento `c[T-1]` tiene como vecinos los elementos `c[T-2]` y `c[0]`.

A tomar en cuenta para mostrar el grafo temporal del automata Regla 30, la interfaz en línea de comando de `JSLinux` tiene 80 caracteres de ancho.

El estado inicial

En el tiempo $t = 0$ el arreglo es inicialmente puesto a cero, salvo por el elemento en posición `T/2` puesto a 1.

Mostrar el automata

Esta parte es la más fácil. Basta recorrer el arreglo; si una célula está en el estado 0, imprimir un carácter espacio, en el caso contrario imprimir un carácter que sea "denso en píxeles" y de forma regular, tal que `X`, `*`, `#`, etc.

Transición

Esta parte es la más difícil :-) Es necesario usar un segundo arreglo para construir el estado al tiempo $t + 1$. Recién cuando se ha construido el estado al tiempo $t + 1$ se puede copiar todo su contenido al arreglo anterior, y seguir repitiendo. Hay varias formas de implementar la transición del autómata, las más sencillas de encontrar son también las más largas de escribir. Por lo cual el desafío acá es hacerlo de manera concisa.

```
begin 755 rule
M?T5,1@$!`O`````````````(``P`!````H($$"#0`````````````#0`(``$
M`````````,```"T````M(`$"+2`!`@4````%`````0````!`````0```+0`
M``"T@`O(M(`$"'P#``!\`P``!O`````0```!````0`0``$"4!`A`E`0(?```
M`(`````&`````!````(```!$!```1)0$"$24!`AO````4`````8````$````
M+VQI8B]L9"UU0VQI8F,N<V\N,`````````````````````0```/"#!`AW
M`O``$@````\`````A`O(+@```!(````6````$(0$"&P````2````'@```""$
M!`@O````$@````!?7W5#;&EB8U]M86EN`&UE;7-E=`!P=71C:&%R`'5S;&5E
M<`!L:6)C+G-O+C````(````%````!`````````````````$````"````
M`P```*R4!`@'`O``L)0$"`<"``"TE`O(!P,``+B4!`@'!`````````````
M`````````````Q[5Z)X8/D\%!44FBN@PO(:)R#!`A15FC$@00(Z"\"``#T
M``!5B>6![*O```"X`````(A%^+@!````B$7YN`$```"(1?JX`O```(A%^[@!
M````B$7\N`````"(1?VX`````(A%_K@`````B$7_N`````"(1;"X1P```%"X
M`````%"-1;%OZ-8!``"#Q`RX`O```(A%U+@`````B(5G____#[Z%9____X/X
M2``^-9````.D4````#[Z%9____XG!@\`!B(5G____Z]>-A6C___\/OHUG____
M`<B-3;`/OI5G____`=$/OA&($(72#XO3````N%@```!OZ'O!``"#Q`3I#@``
M`+@@````4.AA`O``@\O$ZZ&X"@```%#H4O$``(/$!+@`````B(5G____#[Z%
M9____X/X2``^-IP```.D4````#[Z%9____XG!@\`!B(5G____Z]>-1;`/OHUG
M____`<@/OHUG____@\%(@^D!B85@____B<BY2````)GW^8V%:/___P'O#[X(
MP>$"C85H____#[Z59____P'O#[XOP>(!`=$/OH5G____@\`!B8U<____N4@`
M``"9]_F-A6C___\!T(N-7/___P^^$`'1C47X`<@/O@B+A6#___^(".E>____
MN*"&`O!OZ)$```"#Q`3IGO[__\G#````58GE4^@;````@</[$```6UW#58GE
M4^@)````@</I$```6UW#BQPDPP`````````````````````````````````
M``#_-:24!`C_):B4!`@`````_R6LE`O(:`````#IX/____\EL)O$"&@(````
MZ=#_____);24!`AH$````.G`_____R6XE`O(:!@```#IL/___P`````````
M`````````````O```"4````$````2($$"`4````8@OO(!@```,B`!`@*
M````+P````L````O````$O```&R!!`@2````(````!,````(````````````
J`````````````````!$E`O(`````````````````````````````````
`
end
```