Instituto Politécnico Nacional

Escuela Superior de Cómputo

Práctica #1

Sigma

Axel Treviño Palacios
2CM5
31 de Octubre de 2020

1. Objetivo

Programar el universo de las cadenas binarias Σ^n. Dada una 'n' que introduzca el usuario o que el programa lo determine automáticamente. El rango de 'n' debe de estar en el intervalo de [0,1000].

1. El programa debe de preguntar si quiere calcular otra 'n' o no.
2. La salida, expresada en notación de conjunto, debe ir a un archivo de texto.
3. Una segunda salida (archivo de texto) debe concatenar todas las cadenas calculadas en una sola cadena, quitar las llaves, comas y cualquier otro símbolo que no sean 0s y 1s.
4. Del primer archivo de salida, graficar el número de 1s de cada cadena. El eje de las x es la cadena y el eje de las y el número de 1s que tiene esa cadena. Específicamente, calcular y graficar cuando n=23. Al mismo tiempo, calcular la gráfica pero calculando su logaritmo en base 2 y 10 respectivamente.
5. Del segundo archivo de salida, particionar la cadena en subcadenas de longitud 32 y graficar la cantidad de unos de esas subcadenas. Al mismo tiempo, calcular la gráfica pero calculando su logaritmo en base 2 y 10 respectivamente.

2. Codigo

```
1  #include <time.h>
2
3  #include <algorithm>
4  #include <cmath>
5  #include <fstream>
6  #include <iostream>
7  #include <string>
8  #include <vector>
9
10 using namespace std;
11
12 // CONSTANTES
13 const string sigmaFileName = "sigma.txt";
14 const string binarioFileName = "binario.txt";
15 const string graficaFileName = "grafica.dat";
16 const string divisionFileName = "binario32.txt";
17 const string graficaDivisionFileName = "grafica32.dat";
18
19 // VARIABLES
20 ofstream sigmaFile;
21 ofstream binarioFile;
22 ofstream graficaFile;
23 ofstream divisionFile;
24 ofstream graficaDivisionFile;
25
26 // FUNCIONES AUXILIARES
27 int menu() {
28     int respuesta;
29
30     cout << "----------" << endl;
31     cout << "-- Menu --" << endl;
32     cout << "0. Salir" << endl;
33     cout << "1. Insertar limite" << endl;
34     cout << "2. Limite automatico" << endl;
35     cout << "Inserte la opcion: ";
36     cin >> respuesta;
```

```
37
38      return respuesta;
39  }
40
41  // FUNCIONES AUXILIARES
42  vector<bool> dec2bin(long dec, int len) {
43      vector<bool> bits;
44      for (int i = 0; i < len; ++i) {
45          bits.push_back(dec & 1);
46          dec /= 2;
47      }
48      reverse(bits.begin(), bits.end());
49      return bits;
50  }
51  int calcularUnos(vector<bool> binario) {
52      int unos, len;
53      len = binario.size();
54      for (int i = 0; i < len; ++i) {
55          unos += binario[i];
56      }
57      return unos;
58  }
59  void dividirBinario() {
60      ifstream file(binarioFileName);
61      char c;
62      int i = 1;
63
64      while (file.get(c)) {
65          if (i %33 == 0) {
66              i = 1;
67              divisionFile << endl;
68          }
69          i++;
70          divisionFile << c;
71      }
72  }
73
74  // FUNCIONES DE ARCHIVOS
75  void escribirSigma(vector<bool> binario) {
76      int len = binario.size();
77      for (int i = 0; i < len; ++i) {
78          sigmaFile << binario[i] ? '1' : '0';
79          binarioFile << binario[i] ? '1' : '0';
80      }
81      sigmaFile << ',' << endl;
82  }
83  void escribirGrafica(int num, int unos) {
84      graficaFile << num << '\t';
85      graficaFile << unos << '\t';
86      graficaFile << log2(unos) << '\t';
87      graficaFile << log10(unos) << endl;
88  }
89  void escribirGraficaDividida() {
90      ifstream file(divisionFileName);
91      string line;
92      char c;
```

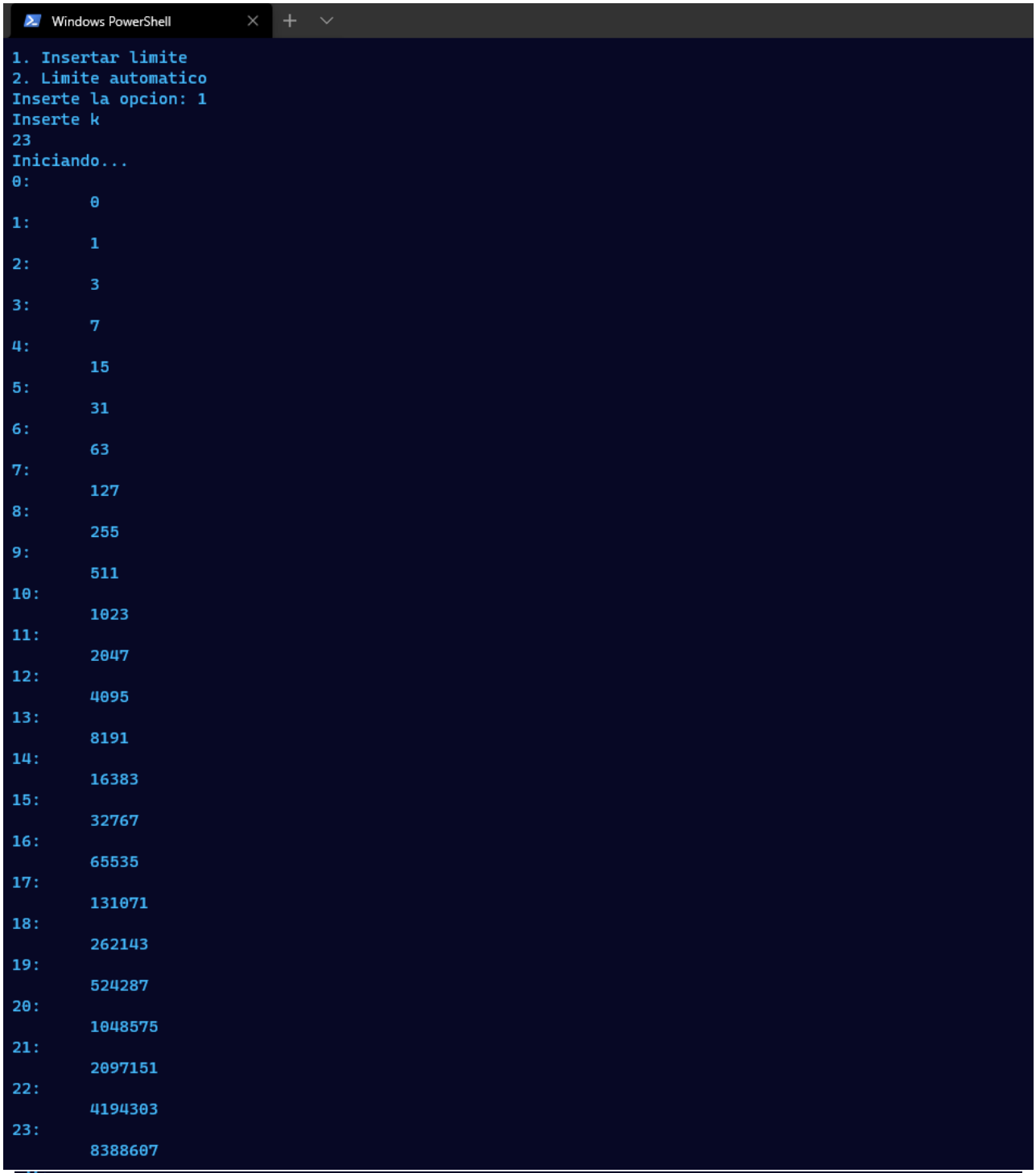
```
93      int i = 1;
94      int unos, len;
95
96      while (getline(file, line)) {
97          unos = 0;
98          len = line.size();
99          for (int i = 0; i < len; ++i) {
100             unos += line[i] == '1' ? 1 : 0;
101         }
102
103         graficaDivisionFile << i << '\t';
104         graficaDivisionFile << unos << '\t';
105         graficaDivisionFile << log2(unos) << '\t';
106         graficaDivisionFile << log10(unos) << endl;
107         ++i;
108     }
109 }
110 void abrirArchivos() {
111     sigmaFile.open(sigmaFileName);
112     binarioFile.open(binarioFileName);
113     graficaFile.open(graficaFileName);
114     divisionFile.open(divisionFileName);
115     graficaDivisionFile.open(graficaDivisionFileName);
116 }
117 void cerrarArchivos() {
118     sigmaFile.close();
119     binarioFile.close();
120     graficaFile.close();
121     divisionFile.close();
122     graficaDivisionFile.close();
123 }
124
125 // FUNCION PRINCIPAL
126 void procesar(int k) {
127     int unos, limite;
128     bool final = false;
129
130     sigmaFile << "{e," << endl;
131
132     cout << "Iniciando...";
133
134     // Calcular las permutaciones
135     for (int i = 0; i <= k; ++i) {
136         cout << endl
137              << i << ":" << endl;
138         limite = pow(2, i);
139         final = i == k;
140
141         for (long j = 0; j < limite; ++j) {
142             cout << "\r\t" << j;
143             vector<bool> binario = dec2bin(j, i + 1);
144
145             escribirSigma(binario);
146
147             if (final) {
148                 unos = calcularUnos(binario);
```

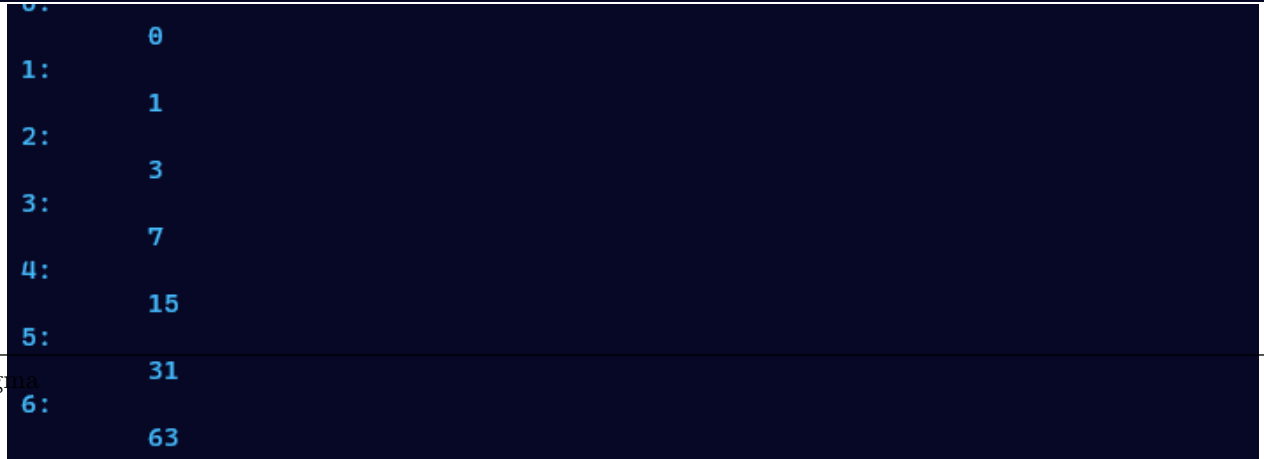
```
149                     escribirGrafica(j, unos);
150                 }
151             }
152         }
153
154         dividirBinario();
155         escribirGraficaDividida();
156     }
157
158     // MAIN
159     int main() {
160         int k;
161         string resultado;
162         int respuesta;
163         bool continuar = true;
164
165         while (true) {
166             respuesta = menu();
167
168             switch (respuesta) {
169                 case 0:
170                     continuar = false;
171                     break;
172                 case 1:
173                     cout << "Inserte k" << endl;
174                     cin >> k;
175                     break;
176                 case 2:
177                     srand(time(NULL));
178                     k = rand() % 1000;
179                     cout << "K = " << k;
180                     break;
181             }
182             if (!continuar) {
183                 break;
184             }
185
186             abrirArchivos();
187             procesar(k);
188             cerrarArchivos();
189
190             cout << endl
191                  << "El resultado esta en los archivos c:";
192         }
193     }
```

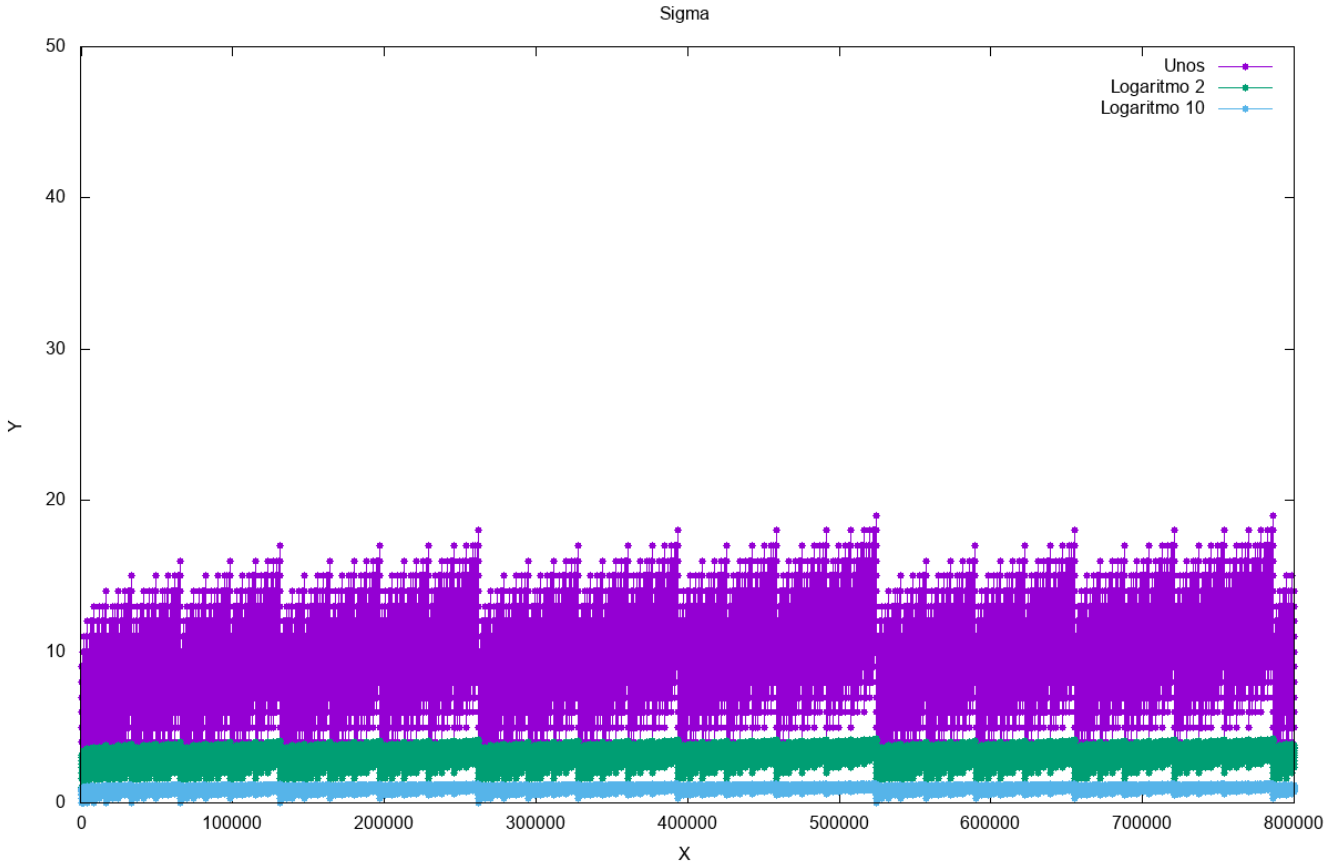
3. Conclusiones

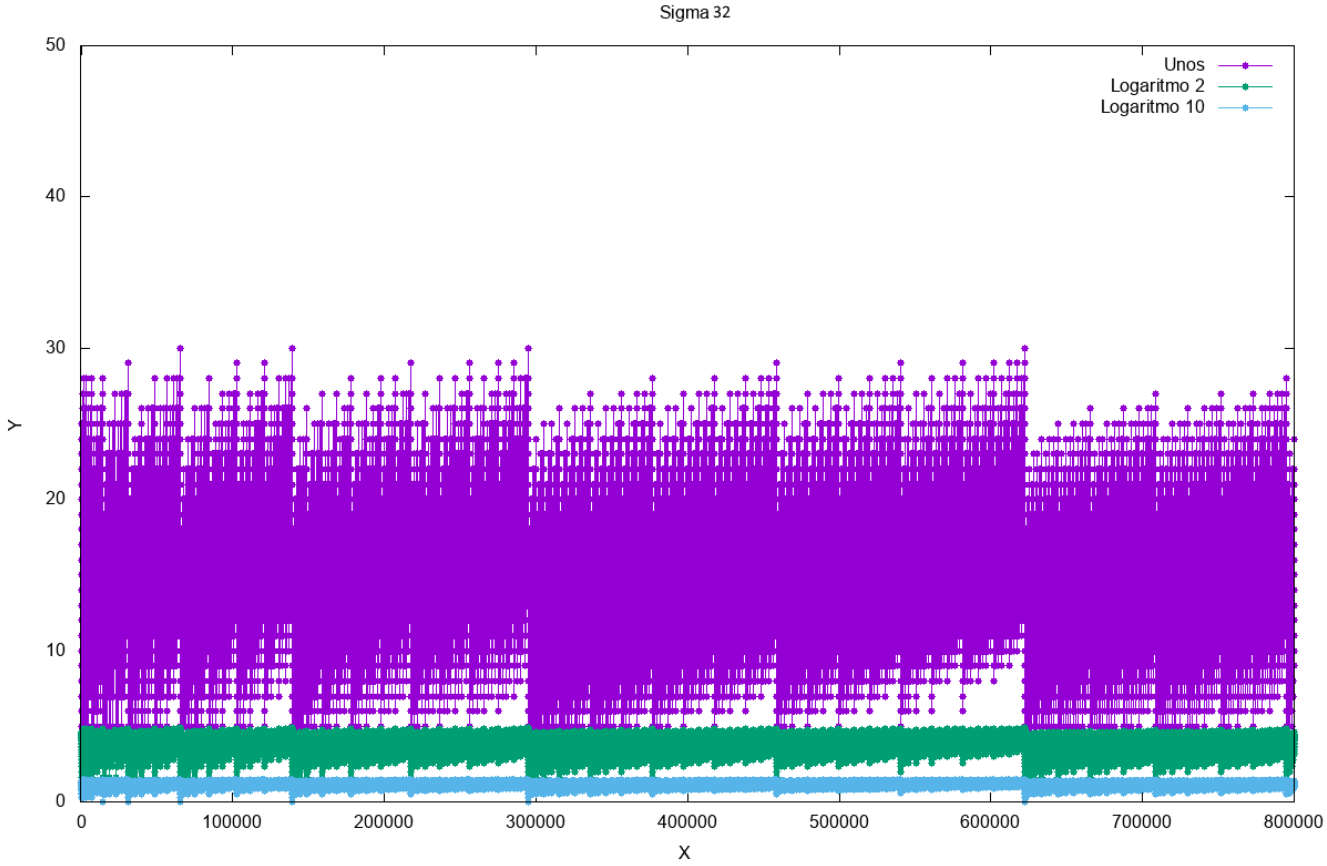
Las computadoras, aunque increíblemente potentes, aun necesitan de nuestra ayuda para no matarse a sí mismas.

4. Resultado


Windows PowerShell
1. Insertar limite
2. Limite automatico
Inserte la opcion: 1
Inserte k
23
Iniciando...
0:
0
1:
1
2:
3
3:
7
4:
15
5:
31
6:
63
7:
127
8:
255
9:
511
10:
1023
11:
2047
12:
4095
13:
8191
14:
16383
15:
32767
16:
65535
17:
131071
18:
262143
19:
524287
20:
1048575
21:
2097151
22:
4194303
23:
8388607


0
1:
1
2:
3
3:
7
4:
15
5:
31
6:
63


Sigma
Unos
Logaritmo 2
Logaritmo 10
Y
X
0
10
20
30
40
50
0
100000
200000
300000
400000
500000
600000
700000
800000


Sigma 32
Unos
Logaritmo 2
Logaritmo 10
Y
X
0
10
20
30
40
50
0
100000
200000
300000
400000
500000
600000
700000
800000