Information Theoretic Analysis: Basics and Application

By: Adel S. El-Atawy

Entropy:

- Randomness
- Uncertainty
- Unexpectedness
- Maximum Achievable Information
- Transferable Energy of a System (related to)

Entropy

Statistical Form (Shannon)

Algorithmic Form (Kolmogorov)

Applications of Entropy Estimation:

Examples:

- Data Complexity Analysis (e.g., intelligent form of life vs. noise)
- Digital Signal Processing (e.g., voice/music separation)
- Medical Applications (e.g., EEG and ECG* entropy related to consciousness, stages of sleep and effectiveness of Anaesthesia, …)
- Biological Applications (e.g., GNOME analysis, junk/gene separation, …)
- Power Consumption of Data Processors (Power consumption increases with increase of data complexity)

* ECG: Electro-Cardiograph
EEG: Electro-Encephalograph

Entropy (Algorithmic)

Information contained in a given data set equals to:
Size of shortest algorithm that can regenerate such data set.

Example:

Seq = <1,1,1,1,1,1...>

Program:

1: print “1”,
2: repeat 1

Seq = <1,3,2,6, 1,3,2,6, 1,3,2,6, 1,3,2,6, …>

Program:

1: print “1,3,2,6”
2: repeat 1

Entropy = Program Length / Sequence length → small value → 0 (approx)

Entropy (Algorithmic) - cont.

Definition:

A random sequence (maximum entropy): a sequence whose shortest generating program is of the same length as the sequence.

Entropy = Program Length / Sequence Length = 1 (approx)

Example:

Seq = <314159 … digits of Pi>, or <2,3,5, prime numbers>
Program:
short program as well ➔ NOT Random

Seq = <7268393… some real random sequence>
Program:
Print "7268393…"

Entropy (Statistical)

Statistical forms of the Entropy H(S) (SHANNON's Entropy):

More Powerful / Accurate ↓

$$H(S) = \log(|S|)$$

$$H(S) = \sum_{S} P(s_i) \log \frac{1}{P(s_i)}$$

$$H(S) = \sum_{S^n} P(s_0 \ldots s_{n-1}) \sum_{S} P(s_n / s_0 \ldots s_{n-1}) \log \frac{1}{P(s_n / s_0 \ldots s_{n-1})}$$

$$= \sum_{S^{n+1}} P(s_0 s_1 \ldots s_{n-1} s_n) \log \frac{1}{P(s_n / s_0 s_1 \ldots s_{n-1})}$$

$$H(S) = \sum_{S^n} P(s_0 \ldots s_{m-1}) \sum_{S} P(s_m \ldots s_{m+e-1} / s_0 \ldots s_{m-1}) \log \frac{1}{P(s_m \ldots s_{m+e-1} / s_0 \ldots s_{m-1})}$$

$H_m^e(S) =$ Entropy Using e symbols, after studying m symbols from History

Probability Theory

Event probability estimation forms:

$$P(A) \cong \frac{n_A}{N} \rightarrow P(A) \cong \frac{n_A + 1}{N + d} \rightarrow P(A) \cong \frac{n_A + \beta}{N + \beta d}$$

Bias term

$$\langle H(S) \rangle = H(S) \underbrace{- \frac{M-1}{2N} + \frac{1}{12N^2} \times \left(1 - \sum_{p_i > 0} \frac{1}{p_i}\right) + O\left(\frac{1}{N^3}\right)}_{\text{Bias term}}$$

where N: processed samples, M: different symbols observed

Entropy Estimation Techniques

- Parametric techniques
  - Known-PDF techniques
  - Unknown-PDF techniques
- Non-parametric techniques:
  - Contextual:
    - Shannon’s Experiment
    - Lempel-Ziv, Ziv-Lempel
  - Non-Contextual
    - Block Entropy & KS Exhaustive listing
    - Gambling
    - Entropy Gradient Technique

Entropy Gradient Technique

The Three Phases of the Technique:

A. Data Collection Phase

- Data sequence processing
- Frequency collecting

B. Calculation Phase

- Probability estimation
- Calculation of single entropy values.

C. Estimation Phase

- Curve Fitting
- Result Extraction
- Conclusion

A. Data Collection Phase:

Data sequence processing

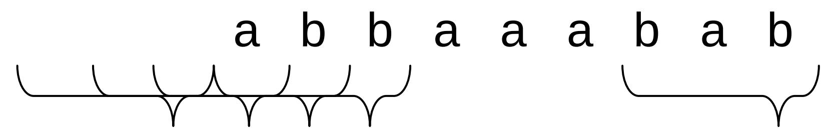
Single pass over the data stream

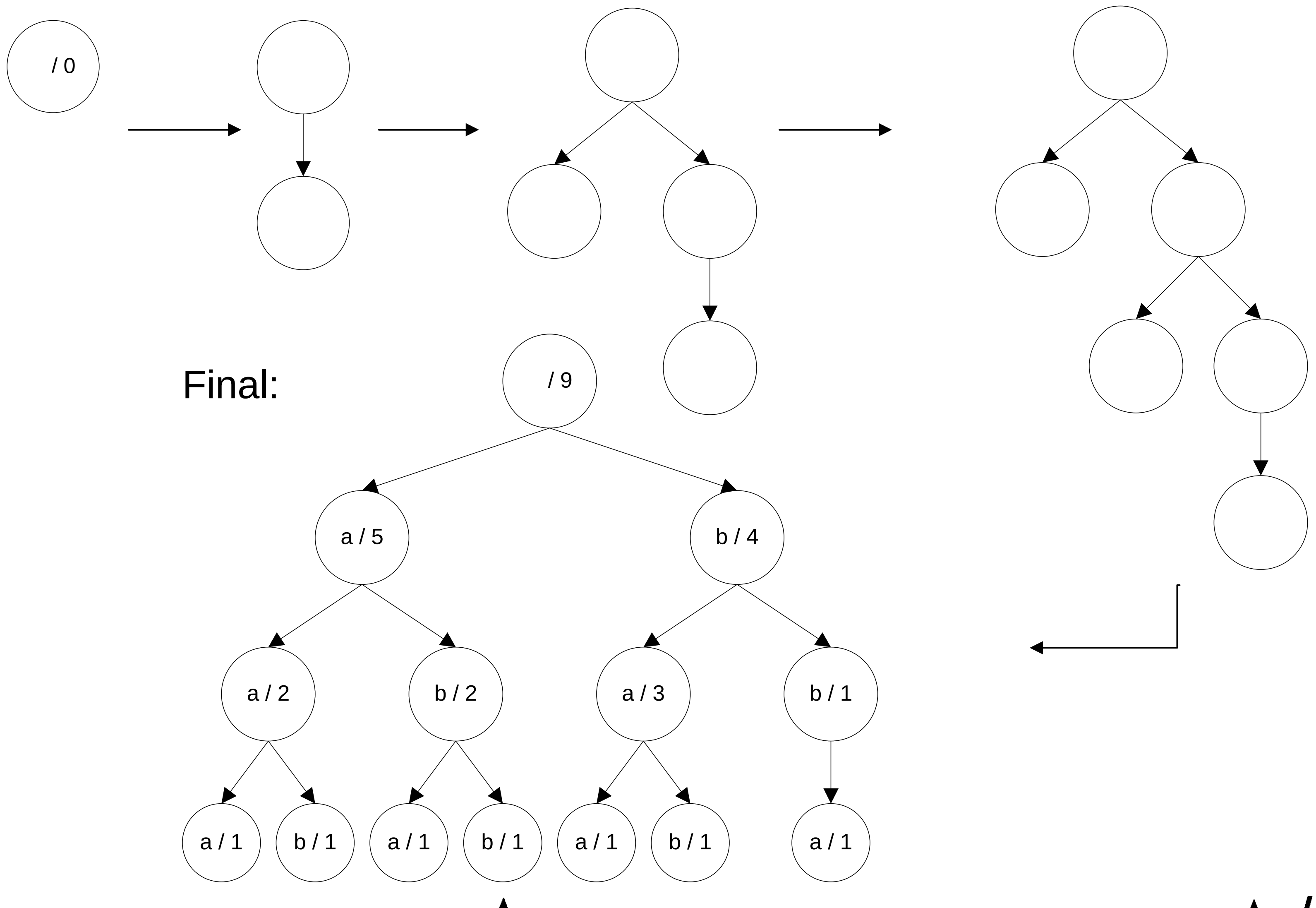
Sliding Window with a constant small size (independent of Stream size)

a b b a a a b a b — Suffix tree

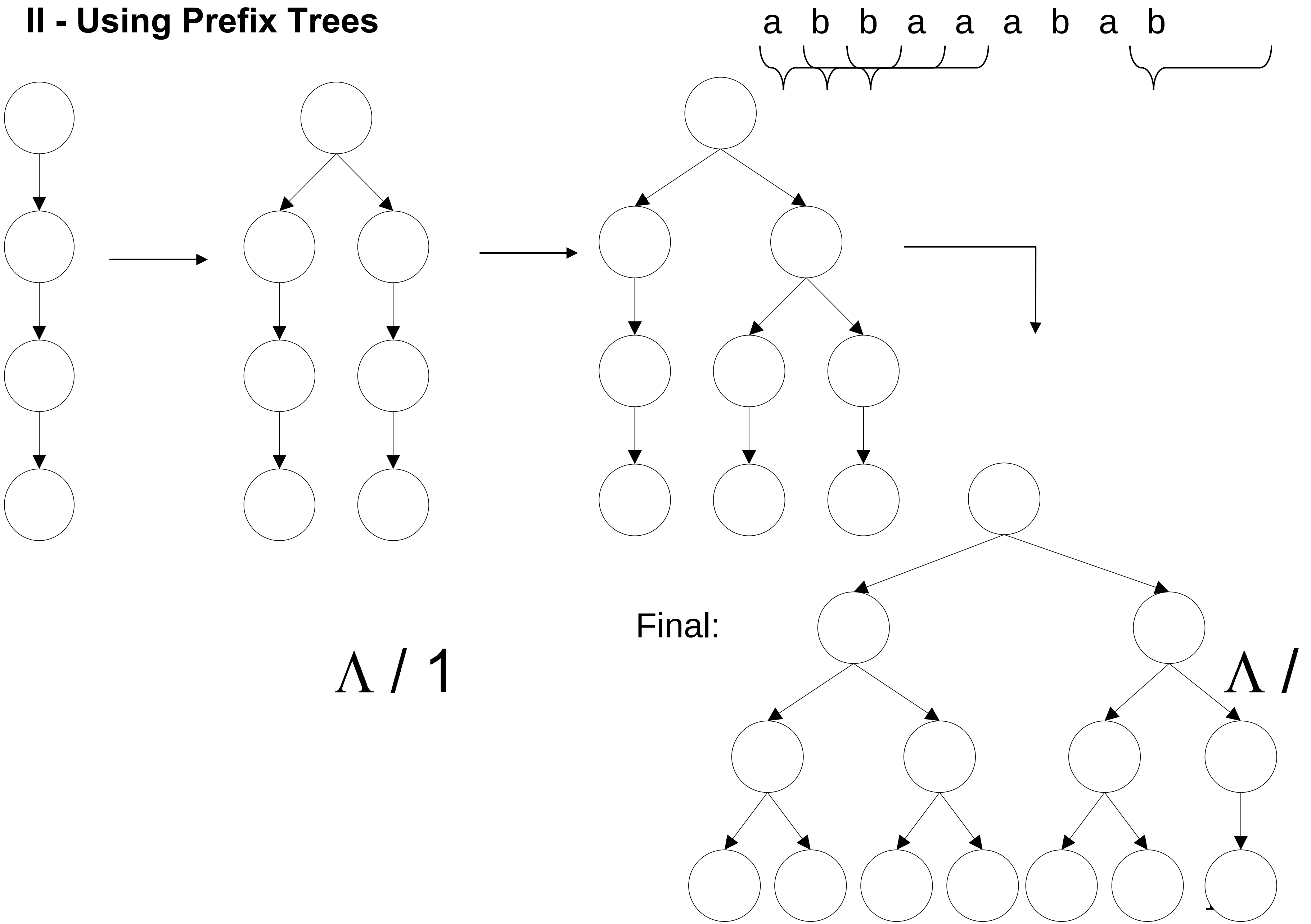
a b b a a a b a b — Prefix tree

I - Using Suffix Trees





II - Using Prefix Trees



Inaccuracies in the frequencies collected

Caused by small sized sequences → Bias

<< First term Solvable >>

Effect of Initial/Final state

<< Can be compensated for >>

The ways to distribute the discrepancies :

➔ Equal Shares, InRatio, Maximize Entropy.

➔ Known/Unknown alphabet size

B. Calculation Phase:

By Traversing the suffix tree (or the prefix tree), we can obtain the following set of values:

Memory

Extension

	0	**1**	**2**	**3**
1	1.0	0.75	0.7	0.55
2	0.8	0.65	0.57	
3	0.7	0.58		
4	0.6			

If Depth of Tree = 4
➔ e+m ≤ 4

i.e., H(e , m) = H_m^e

Note: Shown values are just for demonstration, and are not the result of the analysis of any specific data sequence.

C. Estimation Phase:

- Fitting the values into a suitable Entropy Gradient Function

- Studying the EGF properties

- Extrapolating using the EGF

Entropy Gradient Functions (EGFs):

$$H^*(e,m) = h$$

$$H^*(e,m) = h + \frac{r}{e}$$

$$H^*(e,m) = h + \frac{r}{e+m}$$

$$H^*(e,m) = h + \frac{1}{ae+bm}$$

$$H^*(e,m) = h + (ae+bm)^{-c}$$

$$H^*(e,m) = h + (ae^c + bm^d)^{-1}$$

Fitting Strategy

We can't use direct Least-Square-Error Methods:

- Unequal weight for positive and negative errors
- Unequal reliability of samples (reliability decrease with increasing e and m)

Proposed Optimization Formulation of the Fitting Problem
(The effect of different *e* and *m* values is removed for clarity) :

$$min \quad : \quad error = \sum_{e=1}^{D} \sum_{m=0}^{D-e} \varepsilon^{2}_{e,m,\ v}$$

$$\varepsilon_{e,m,v} = \begin{cases} \lambda(H^{*}(e,m)\big|_{v} - H(e,m)) & , H^{*}(e,m)\big|_{v} < H(e,m) \\ (H^{*}(e,m)\big|_{v} - H(e,m)) & , H^{*}(e,m)\big|_{v} > H(e,m) \end{cases}$$

Where: $\lambda > 1$

Applications and Observations

New Applications:

Studying Finite Memory Machines

Number Properties

Test Cases For Technique Verification:

Exponentially-Distributed Sources

Locality Graphs

English Text

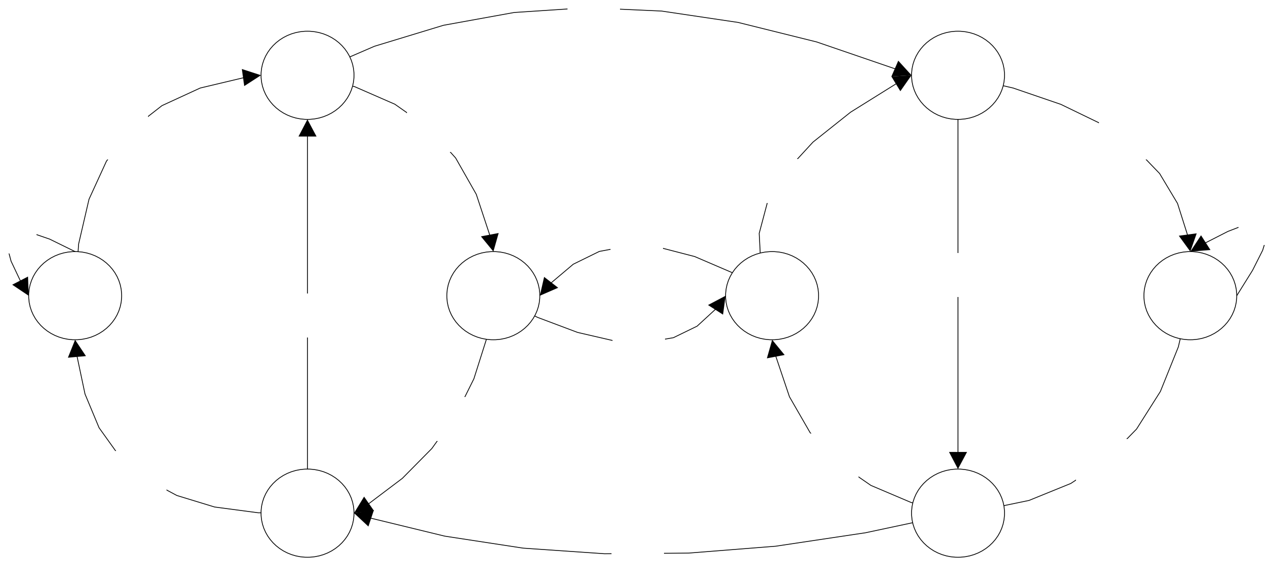
Finite Memory Machines (1/5):

Identifying the memory depth of a Finite memory Machine:

A Finite Memory Machine with memory *M*, is a machine that given the previous M inputs/outputs pairs, you can identify the current state (or given the next input you can identify the next output)

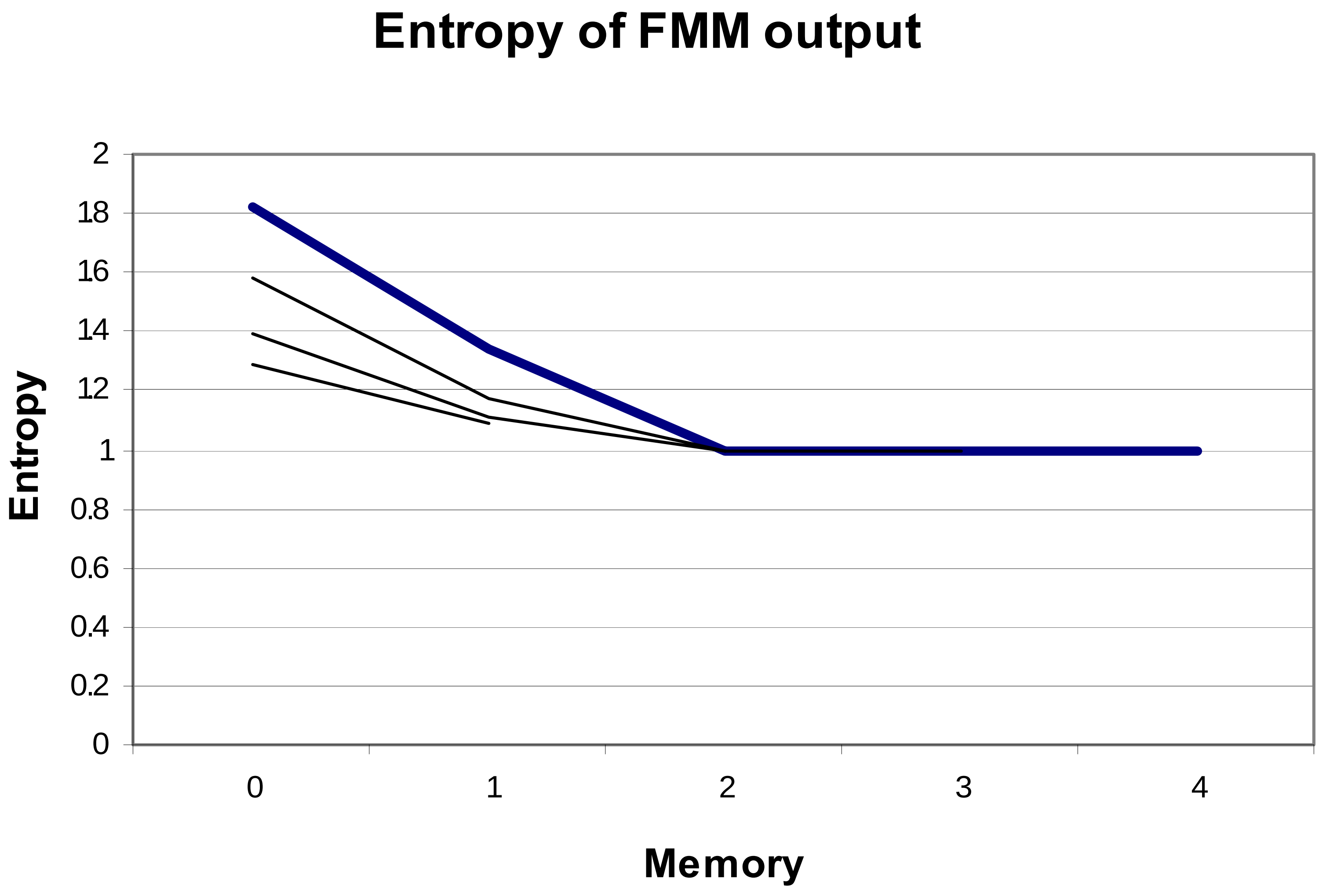
1. Generate a test sequence (X) (a pseudo-random sequence is enough)
2. Input the sequence X to the Machine.
3. Save each input x, with its output y: <x, y> = z
4. Calculate the Entropy of X, and Z. (with different memories)
5. The memory at which H(X) = H(Z) is the machine's M.

Finite Memory Machines (2/5):

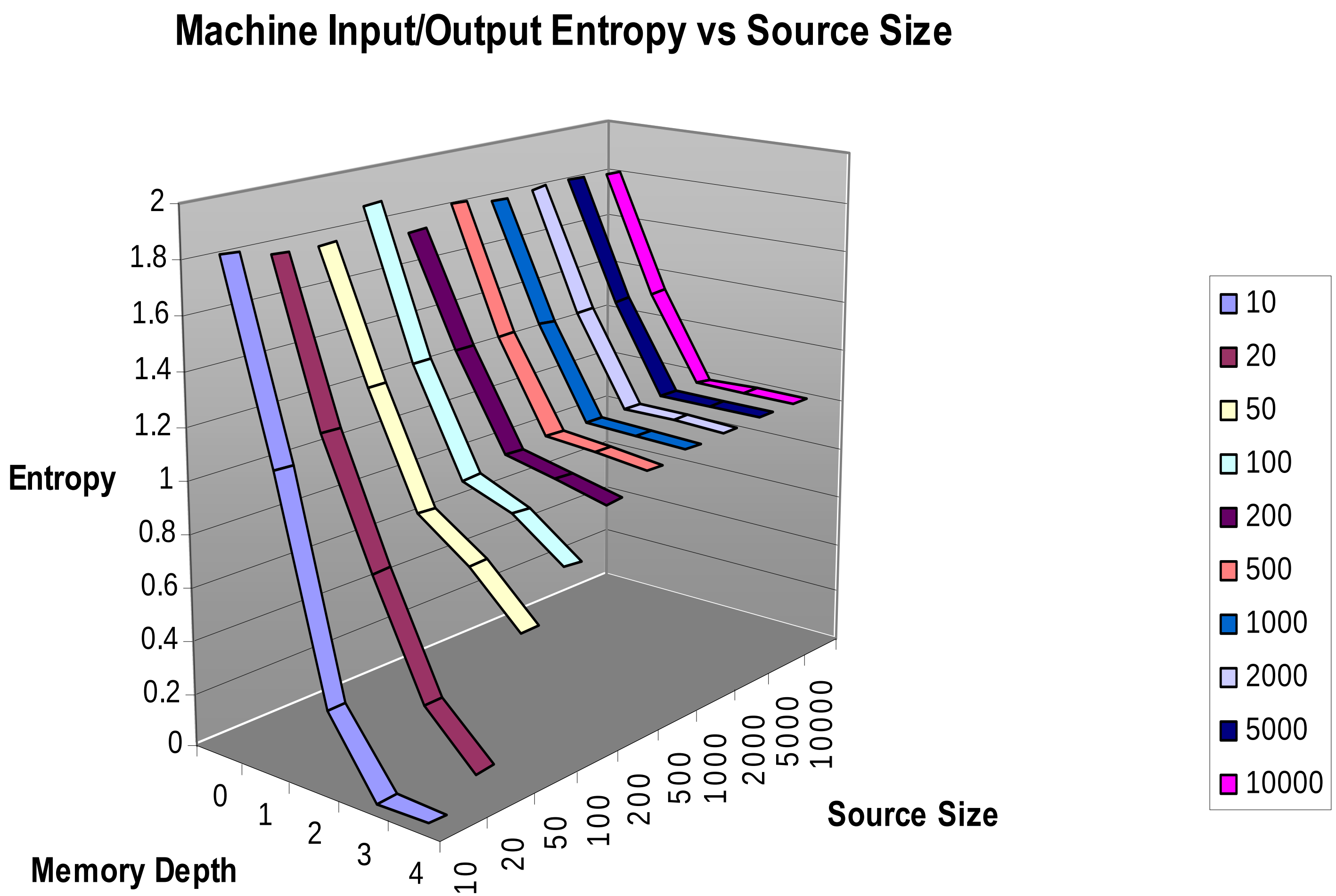


Entropy Drops with increasing memory till it reaches (with feasible noticeable precision) the entropy of the input test signal.
In the example: m = 2, while states appears to need 3 bits of memory.

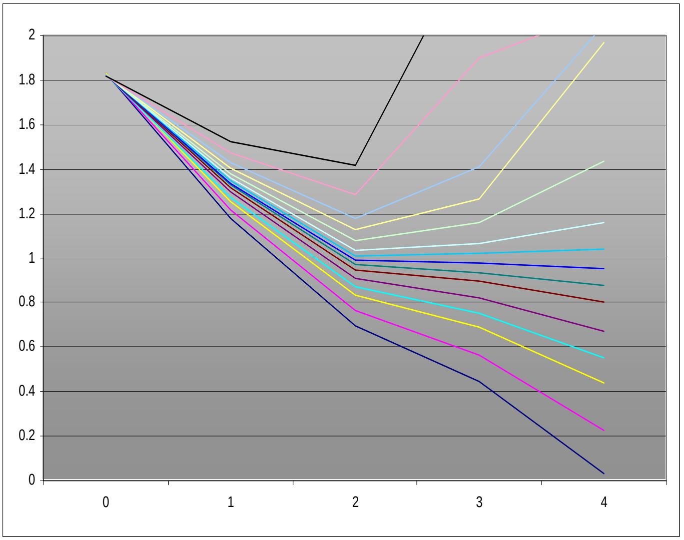
Finite Memory Machines (3/5):

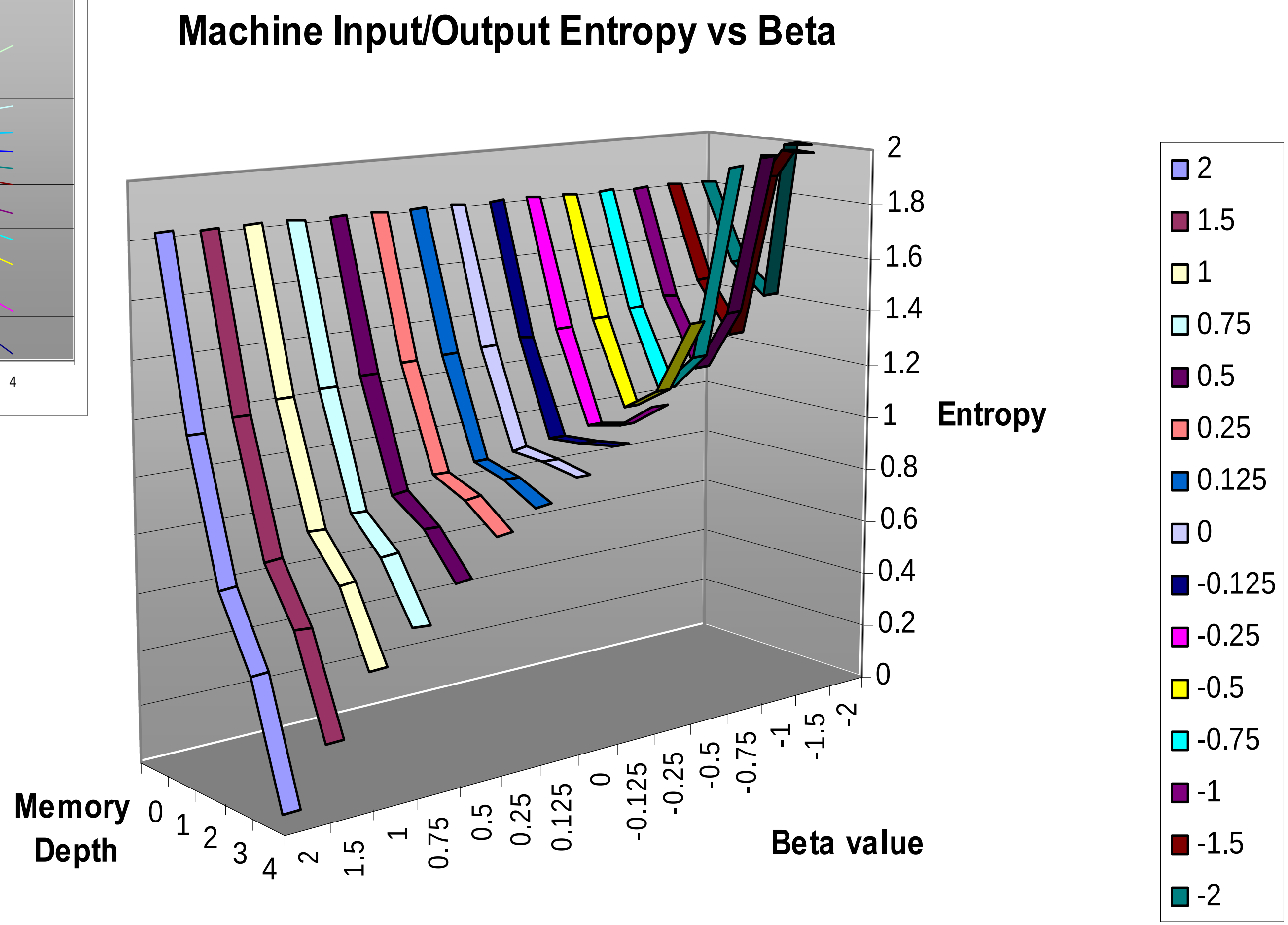


Finite Memory Machines (4/5):



Finite Memory Machines (5/5):





Number Properties (1/7)

Statistical relations can be established between :
the entropy of the number's representation and some of the number's properties

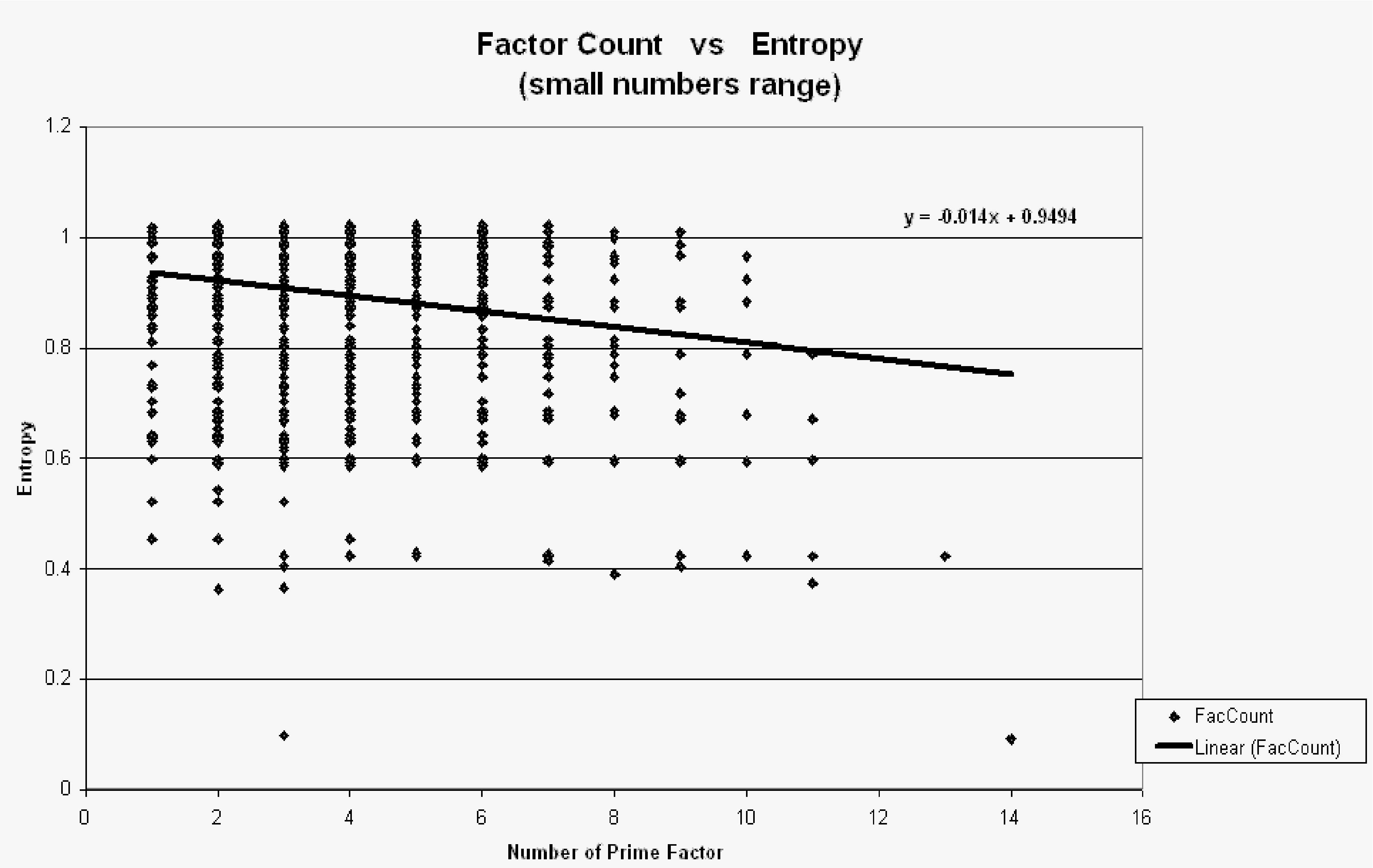
Number = 2655_{10} = 101001011111_2

2655 = 5 * 531

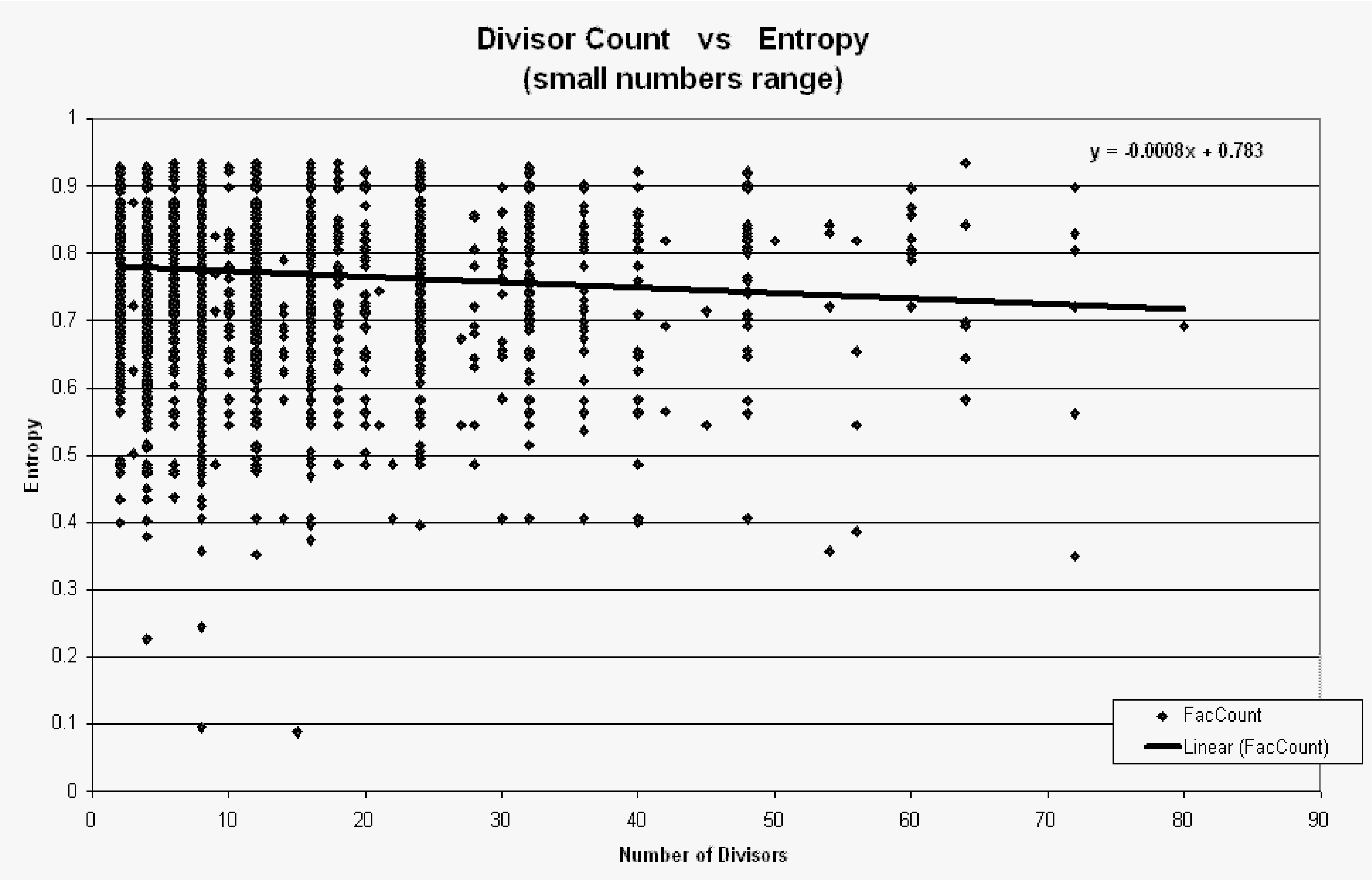
Note: **101**00**101**1111

•**Factor Count**	→	**inverse**
•**Divisor Count**	→	**inverse (weaker)**
•**Distinct Factor Count**	→	**direct (very weak)**
•**Minimum Factor**	→	**direct**
•**Average Factor**	→	**direct (stronger than MinFac)**
•**Maximum Factor**	→	**direct (stronger than AvgFac)**

Number Properties (2/7)

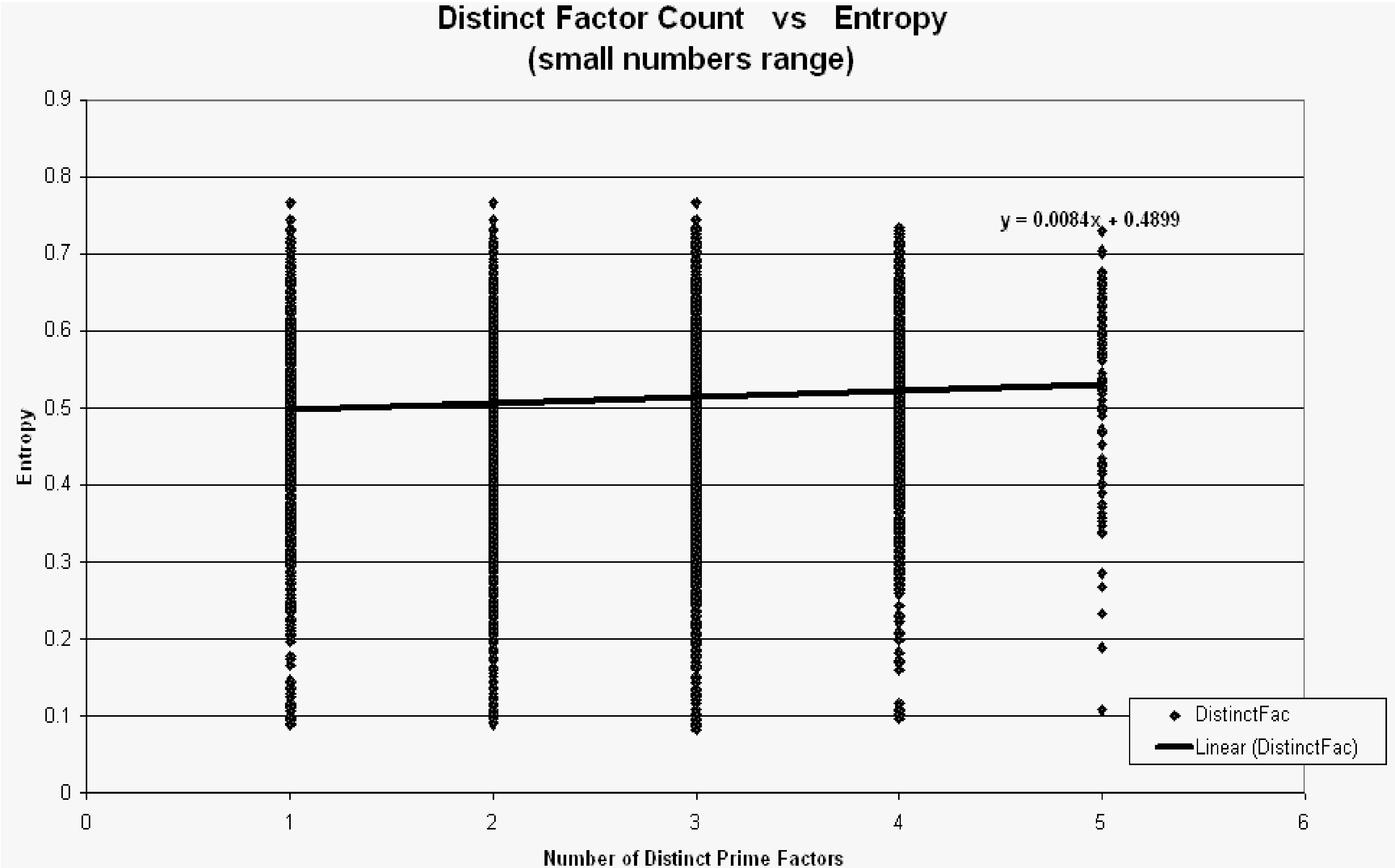


Number Properties (3/7)

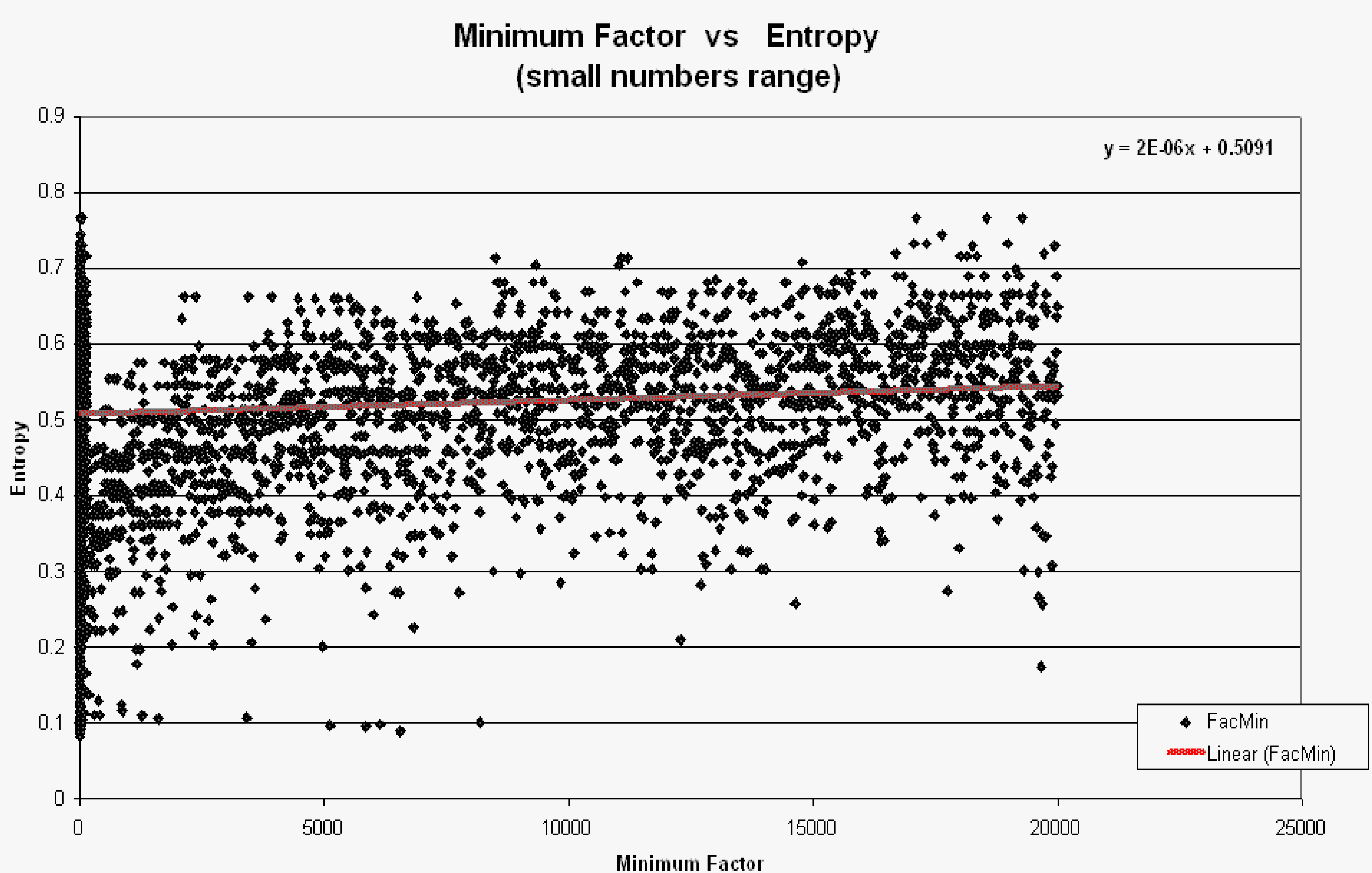


Applications and Observations

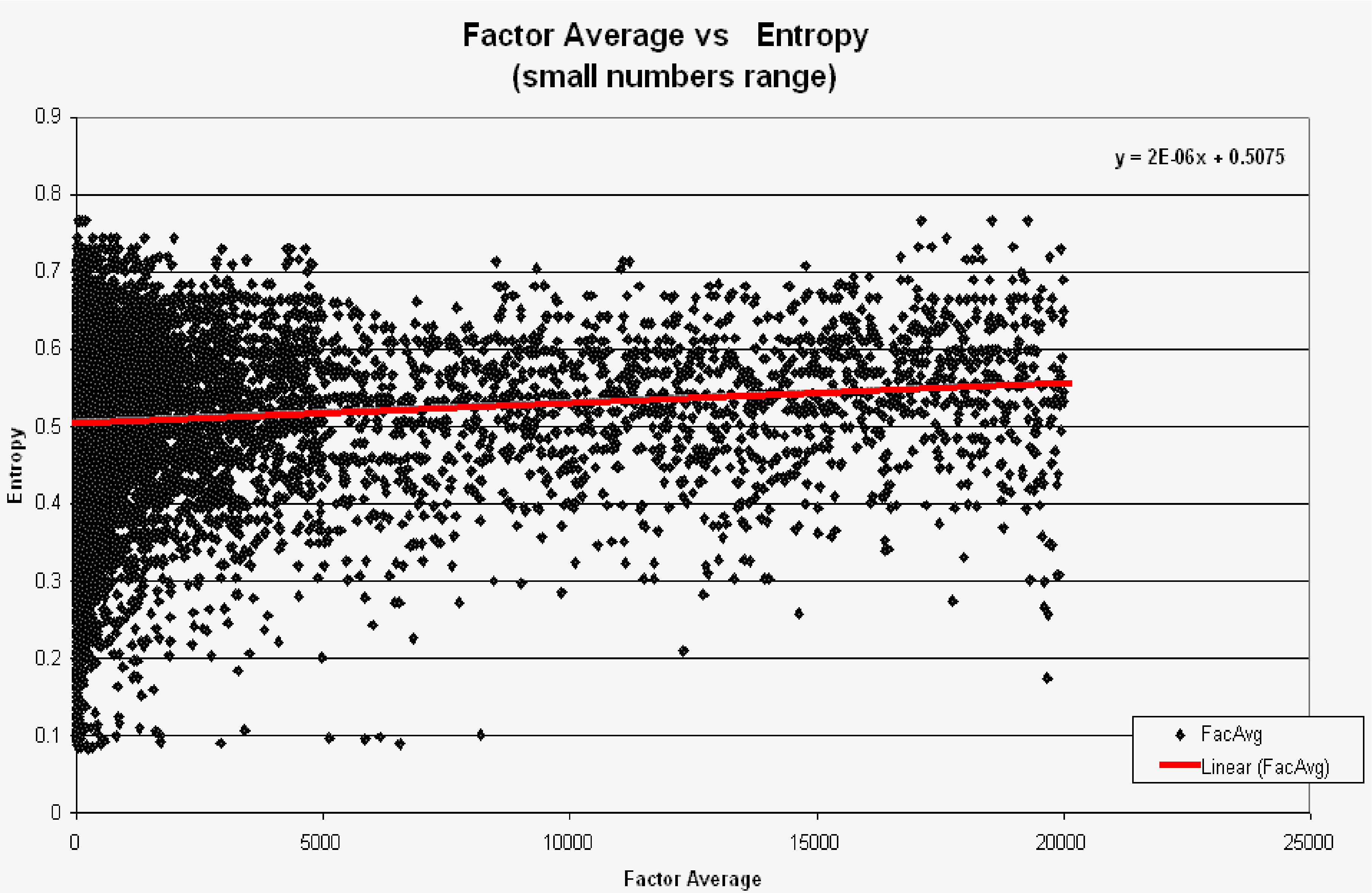
Number Properties (4/7)



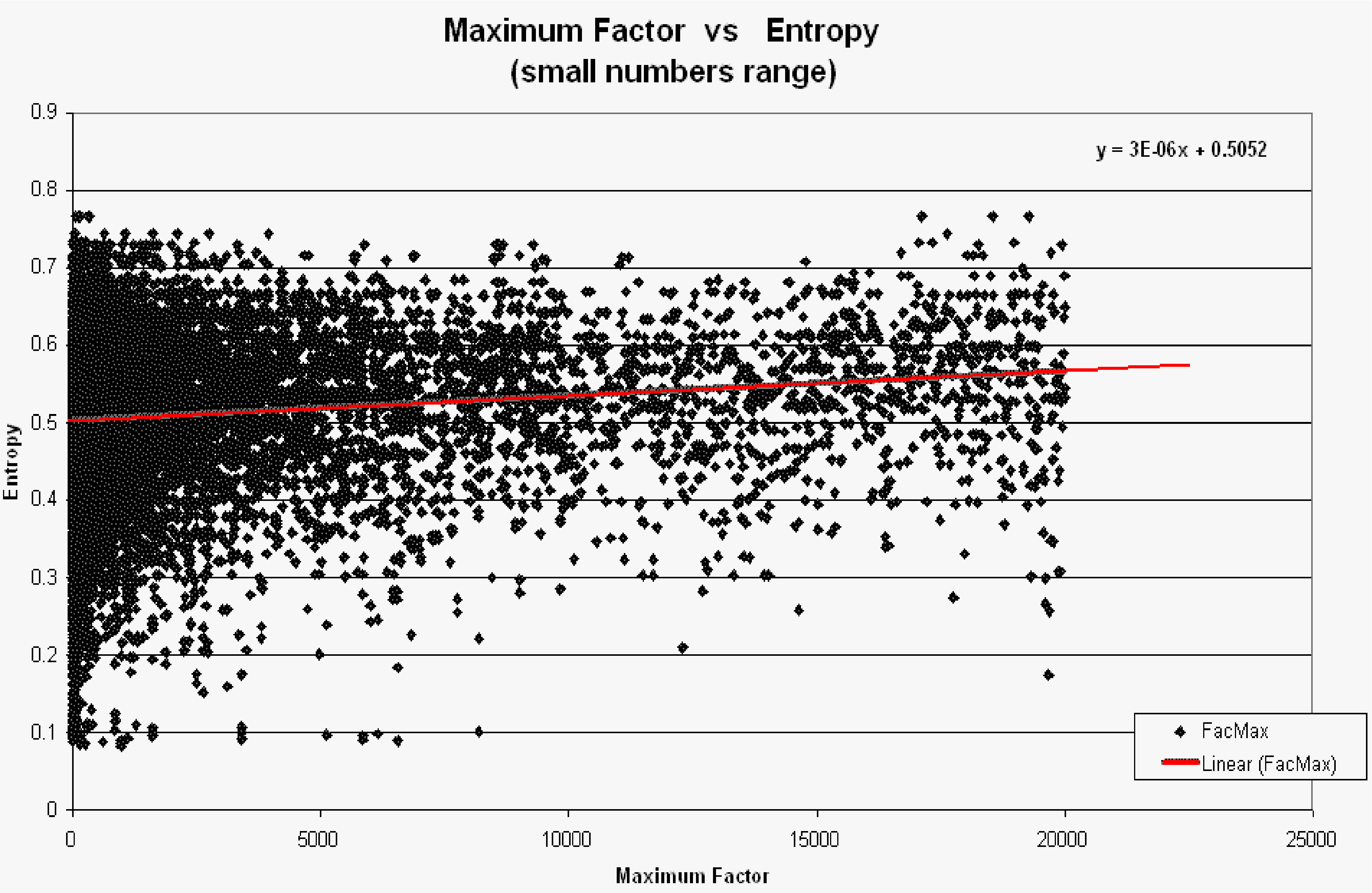
Number Properties (5/7)



Number Properties (6/7)



Number Properties (7/7)



Special Sources:

Exponential Distribution (1/4):

$$P(x=i)=\begin{cases} 2^{-i} & ,0<i<n \\ 2^{-(n-1)} & ,i=n \\ 0 & ,otherwise \end{cases}$$

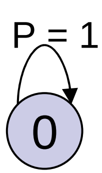
$H(s) = 2 – (1/2)^{n-2}$

Example:

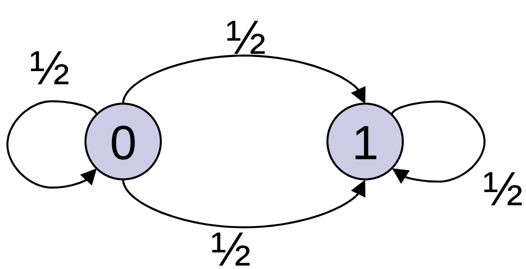
For n=3, P(A)= ½ , P(B) = ¼, P(C) = ¼

Entropy = 2- (½)$^{(3-2)}$ = 1.5 bits/symbol

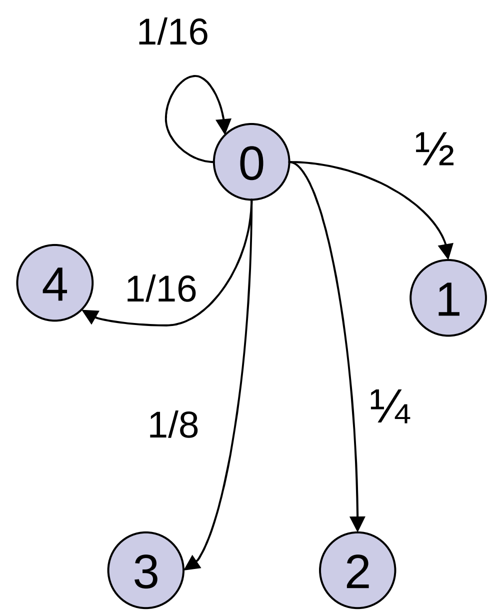
One way-Random walk on graphs:



n=1



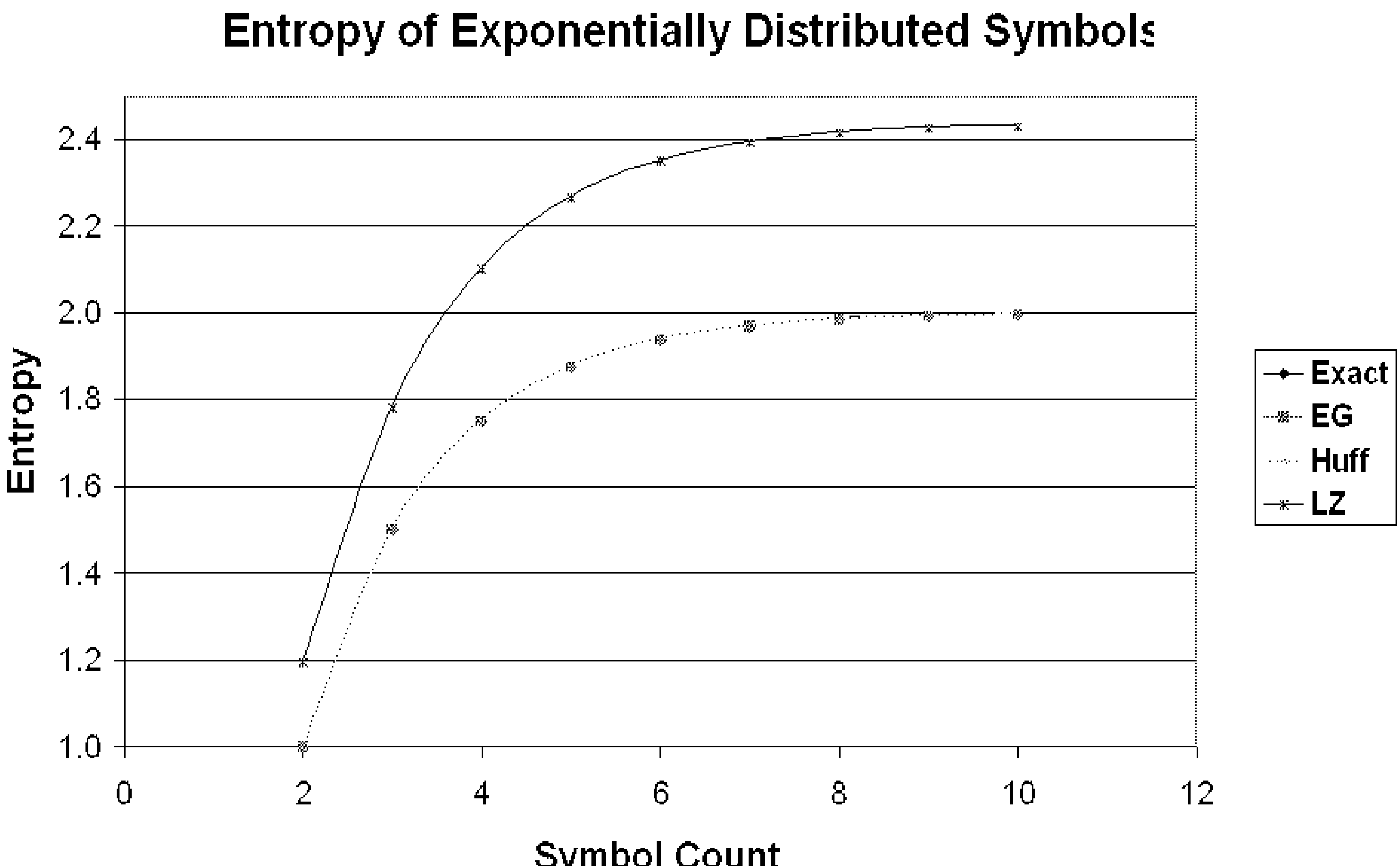
n=2



n=5 **(only one node transitions are shown)**

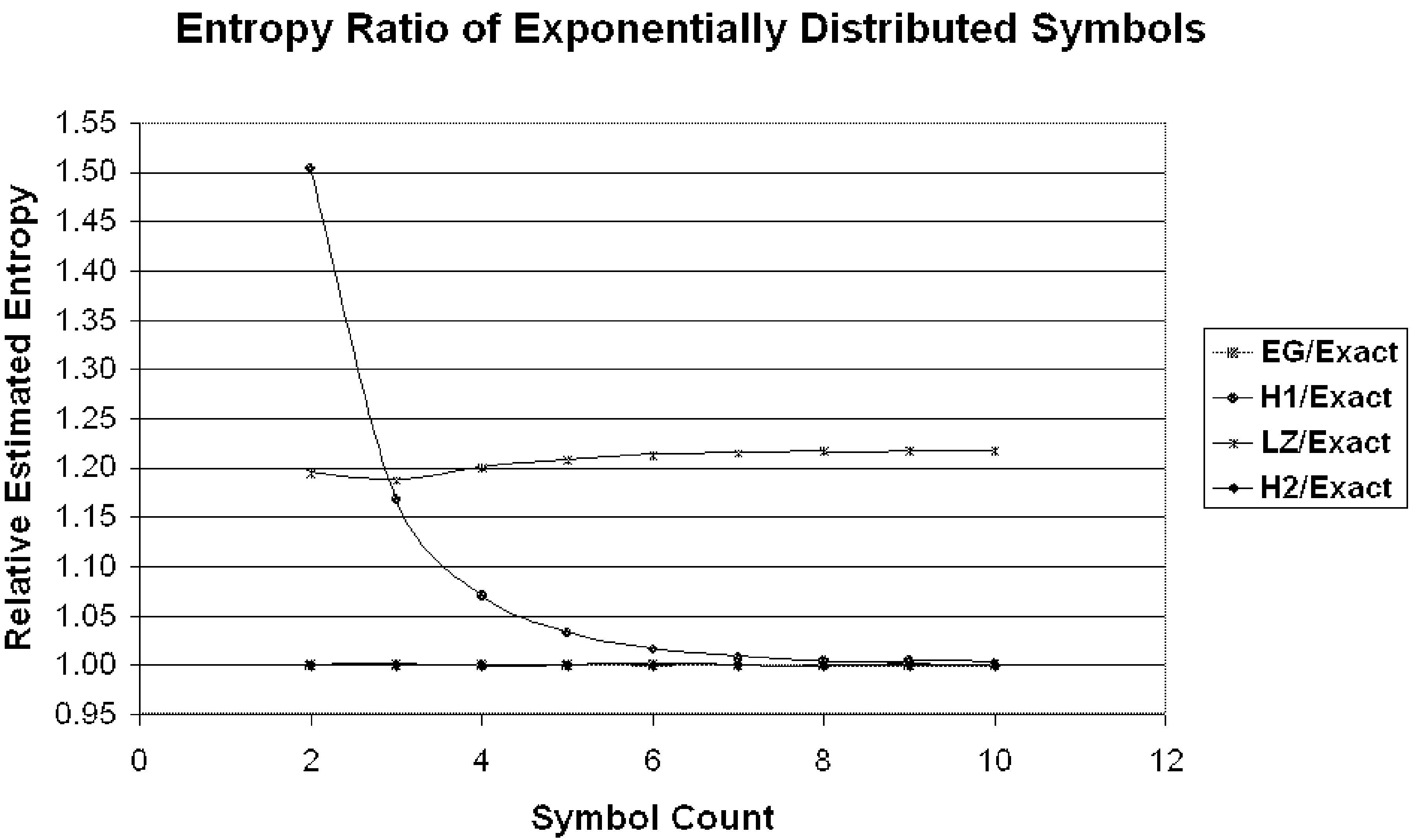
Special Sources:

Exponential Distribution (2/4):



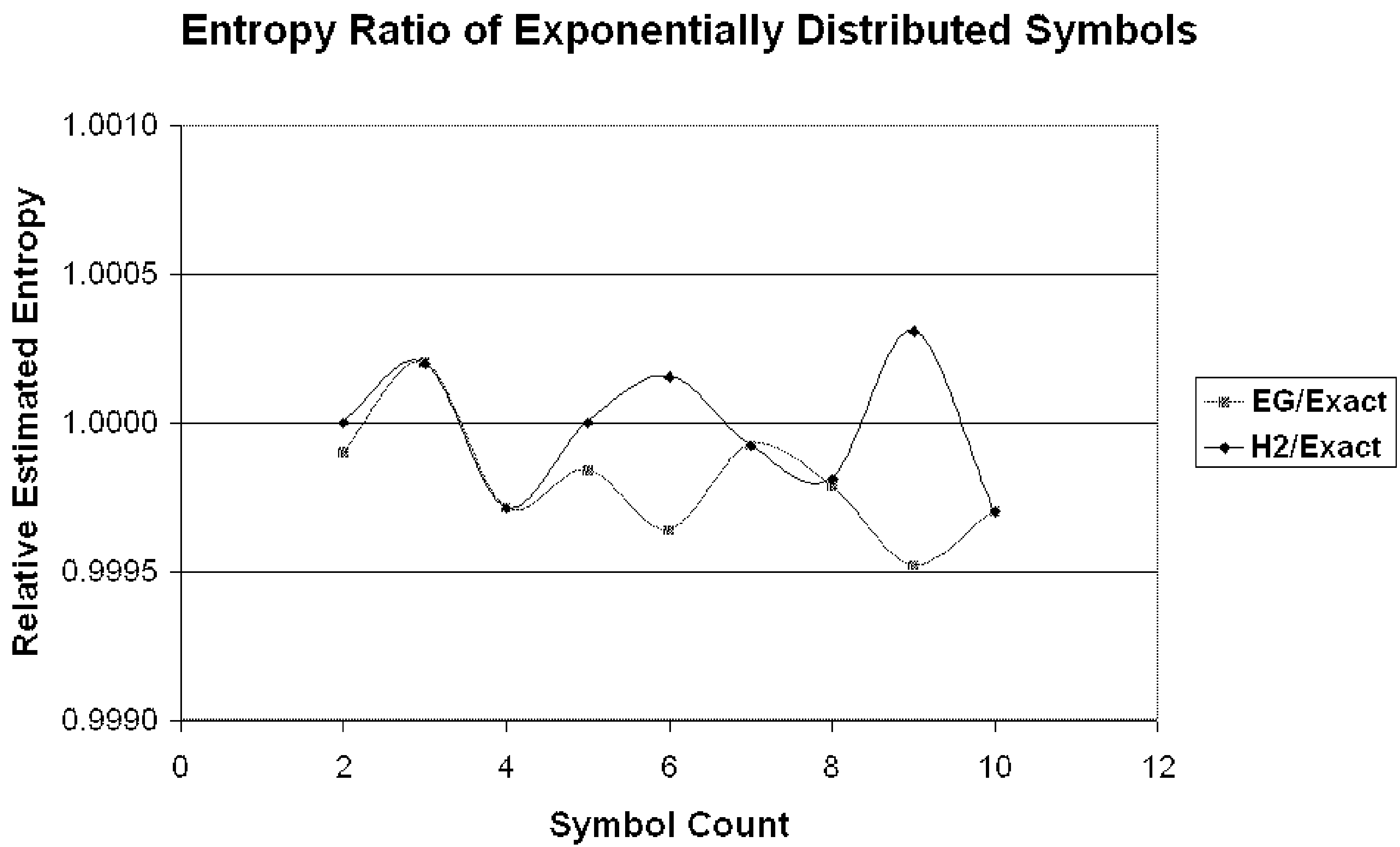
Special Sources:

Exponential Distribution (3/4):



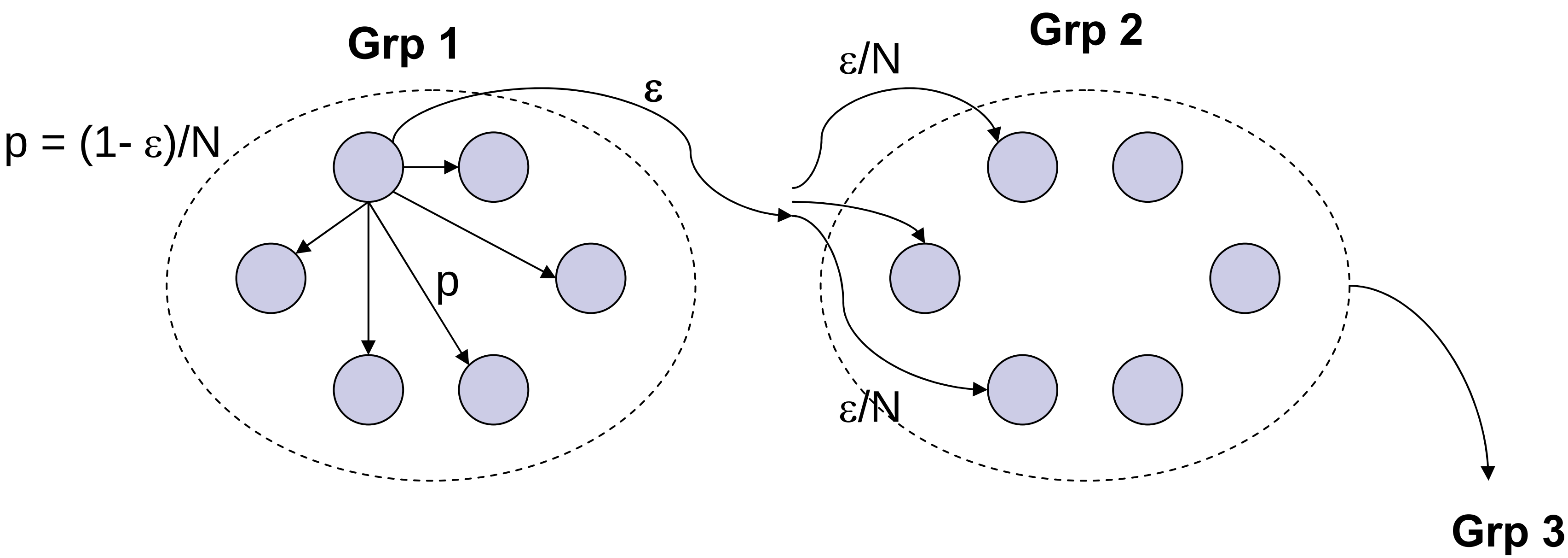
Special Sources:

Exponential Distribution (4/4):



Special Sources:

Locality Referenced States :

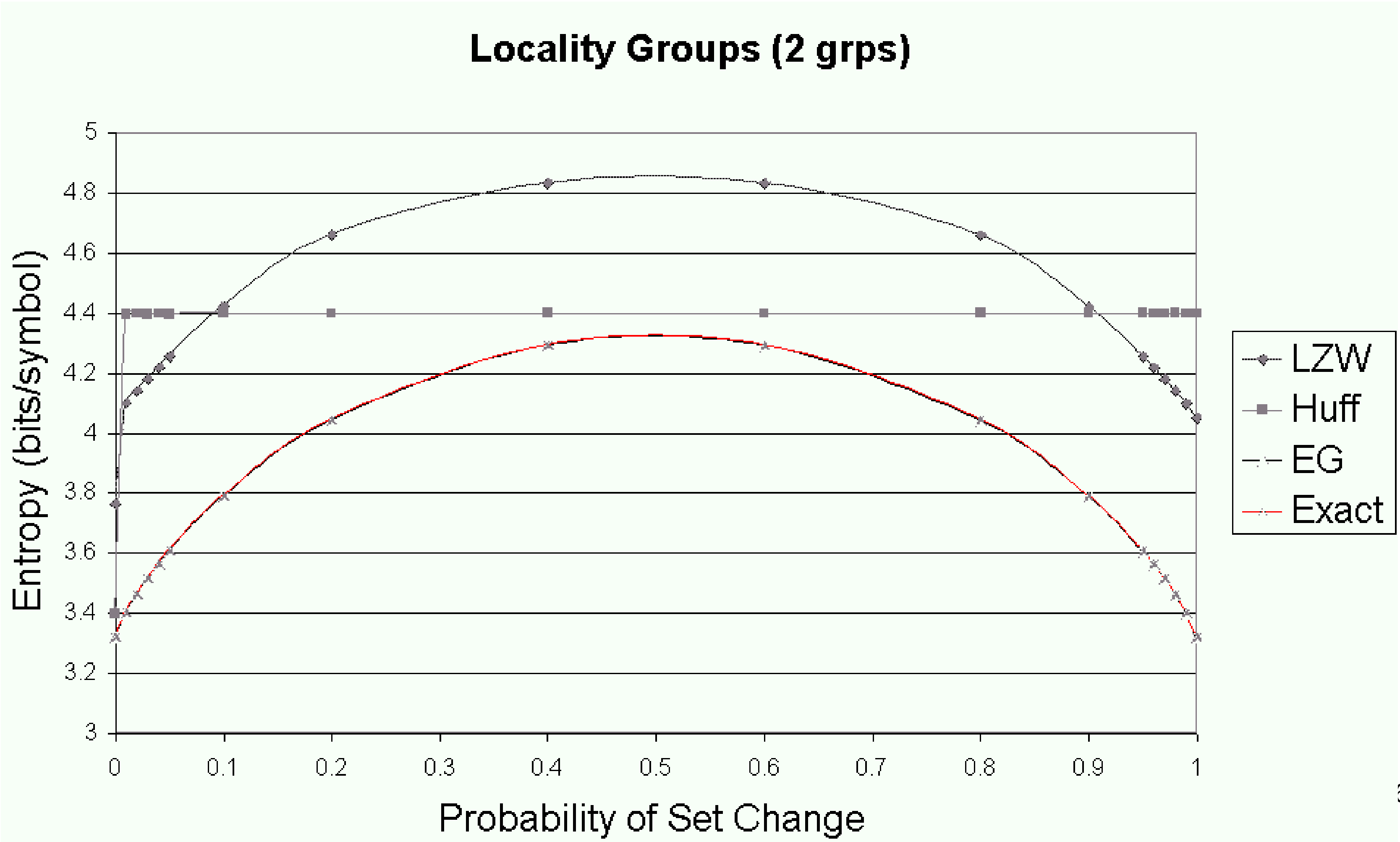


Each Symbol X = <n,m>
n: node number
m: group Number

H(S) = H(choosing a node) + H(choosing group)

Special Sources:

Locality Referenced States :



Special Sources:

Set of Bernoulli Trials (Binomial distribution):

Single Bernoulli Trial:

$$P(x = True) = p$$

$$P(x = False) = 1 - p$$

$$H = -[p \log(p) + (1-p)\log(1-p)]$$

Special Sources:

Set of Bernoulli Trials (Binomial distribution):

"N" Bernoulli Trials:

$$P(x=i)=\begin{cases} p^i(1-p)^{N-i}\binom{N}{i}, & 0\le i\le N \\ 0 & ,otherwise \end{cases}$$

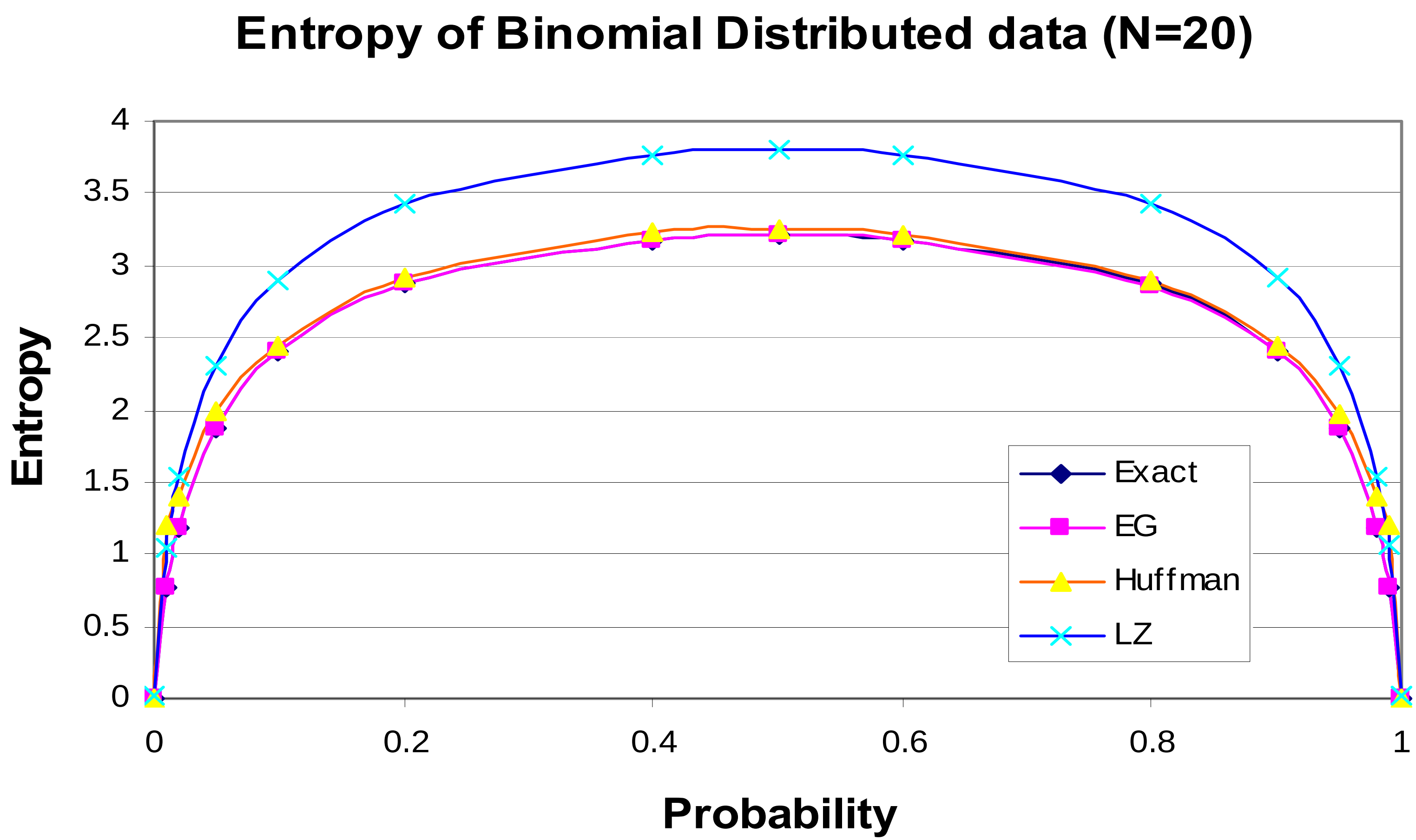
$$H=-\sum_{i=0}^{N} p^i.\overline{p}^{N-i}.\binom{N}{i}.\log(p^i.\overline{p}^{N-i}.\binom{N}{i})=$$

$$E[i].\log(p)+E[N-i].\log(\overline{p})-E[\log\binom{N}{i}]=$$

$$N.H(p)-Selection\ Information$$

Special Sources:

Set of Bernoulli Trials (Binomial):



Conclusion:

- A new entropy estimation technique is presented: **"Entropy Gradient"**.
    - Makes use of the gradient of entropy for various memories and extensions for the estimation.
    - Efficient use of collected information.
- Introducing two new applications:
    - Studying Finite memory Machines
    - Studying the Properties of Numbers.

That’s it. . . .

< < Thank You All > >