The entropy function for non polynomial problems and its applications for turing machines.

Matheus Santana

Harvard Extension School

santanalima.matheus@gmail.com

Abstract

We present a general process for the halting problem, valid regardless of the time and space computational complexity of the decision problem. It can be interpreted as the maximization of entropy for the utility function of a given Shannon-Kolmogorov-Bernoulli process. Applications to non-polynomials problems are given. The new interpretation of information rate proposed in this work is a method that models the solution space boundaries of any decision problem (and non polynomial problems in general) as a communication channel by means of Information Theory. We described a sort method that order objects using the intrinsic information content distribution for the elements of a constrained solution space - modeled as messages transmitted through any communication systems. The limits of the search space are defined by the Kolmogorov-Chaitin complexity of the sequences encoded as Shannon-Bernoulli strings. We conclude with a discussion about the implications for general decision problems in Turing machines.

Keywords: Computational Complexity, Information Theory, Machine Learning, Computational Statistics, Kolmogorov-Chaitin Complexity; Kelly criterion

1. Introduction

Consider a Shanon-Bernoulli process P defined as a sequence of independent binary random variables $X_1, X_2, X_3 \dots X_n$. For each element the value can be

0 or 1. All values have a given probability p . Thus the process is a sequence of
 5 independent and identically distributed Bernoulli trials. Let $H(P)$ be the binary
 entropy function defined as the entropy of a Bernoulli process with probability
 p of one of two outcomes. A state is the representation (i.e. encoding) of a
 choice of the possible values for p . For any given trial there is no information
 about previous executions and future outcomes. Each execution of the process is
 10 stateless [1] [2]. Let a function $g(X)$ be an indexing function. This function is a
 utility function used to represent a preference of ordering and is used to compare
 if alternative state A is preferable against alternative state B . If the probability
 density function for $g(X)$ is an instance of a normal density function than the
 average of a given set of observations of a random variable - with finite mean
 15 and variance - is also a random variable that has a distribution that converges to
 a normal distribution as the number of samples increases. The entropy $H(g(X))$
 for the function $g(x)$ from a random variable X can not be greater than the
 entropy of the random variable X . Let $Y=g(x)$. If the covariance of X and Y is
 not equal to zero than X and Y are correlated.

20 A Bernoulli scheme is a special case of a Markov chain and it's a general-
 ization of the Bernoulli process to more than two possible outcomes from an
 independent random variable. A Markov source is defined as a information
 source created by a stationary finite Markov chain. The Kolmogorov complex-
 ity of a string is the length of the shortest computer program executed by a
 25 Turing Machine that produces the sequence as output and halts. It's the mea-
 surement of the computational resources required to define this string. It can
 be proved that the Kolmogorov complexity of any string can not be larger than
 the expected length of the Bernoulli sequence itself. The entropy of the Markov
 information source is related to the Kolmogorov complexity. From "Kolmogorov
 30 complexity" in Wikipedia [3] [4]: "the Kolmogorov complexity of the output of
 a Markov information source, normalized by the length of the output, converges
 almost surely (as the length of the output goes to infinity) to the entropy of the
 source."

Let $E(Y)$ be the expectation of the $H(Y)$. A bit of information is the amount

35 of entropy encoded in an event with two possible outcomes and even odds. The mutual information between two random variables X and Y can be used to optimize the growth of the utility function $g(X)$ over many trials/executions. This is the "information gain" of a probability distribution for X given the value of Y relative to a given predefined a priori distribution (like the normal distribution)

40 - or stated probabilities - on random variable X . Let $K(E(Y), H(Y))$ be the maximization of the expected value of the logarithm of the entropy of the utility function Y , which is equivalent to maximize the expected geometric growth [5]. This fraction is the level of uncertainty between current and future states of X relative to $E(Y)$. In other words this value represents the intensity of the

45 "side information" measured from each element of the Bernoulli sequence. A program executing in any Universal Turing Machine can use this information to reduce entropy by adjusting the error rate in the long run. At each iteration the program must evaluate the level of uncertainty and decide if to halt or not. This implications contradicts the standard definition of the halting problem in which

50 it proves that there is no general algorithm to solve the halting problem for all possible program-input pairs. The problem is in determining from an arbitrary description of an arbitrary program and an input whether the program will halt or run forever.

By analyzing the entropy of the logarithm of the utility function $g(X)$ relative to the probability density function of a random variable X we can extract

55 useful information about the overall distribution of possible output values in a given sample. Any program p_n can use this knowledge to determine whatever programs halt for a given subset of program-input pairs symbols.

2. Thesis statement

60 The statistical analysis of the entropy function of $H(Y=g(X))$ and its correlation with random variable X can be used by any program p_n to decide the limits in which cases the near-optimal values satisfy the halting condition. This decision must have statistical significance (under a given degree of freedom)

and reduce the uncertainty about the possible states of values from random
 65 variable Y. In section 3.1 we describe the normal distribution function and the
 Shapiro-Wilk test of normality. In section 3.2 we describe the Shannon entropy
 function for Bernoulli sequences. In section 3.3 we discuss the optimization of
 the logarithm of the utility function of a random independent variable using
 the Kelly criterion and the Chi-square distribution. In section 3.4 we present
 70 the Indexing binary function for classification of states in decision problems.
 In section 3.5 we present an information theory interpretation applied to non
 polynomial problems such as the Traveling Salesman Problem. The conclusion
 summarizes the discussion and the implications for the computational limits of
 Turing Machines for a finite alphabet with a given mean and variance.

75 3. Methods and Background

3.1. Sufficient Statistic

3.1.1. Normal distribution

The normal distribution (or the normal density function) is a continuous
 probability distribution for a random variable. The parameter value are the
 80 mean (or expected value) of a given real number and the standard deviation.
 The domain is $(-\infty, +\infty)$. Its defined by the formula show in figure 1:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

Figure 1: Formula for the normal distribution

From "Calculus Applied to Probability and Statistics" [6] the properties of
 the gaussian-normal curve are defined in figure 2::

Properties of a Normal Density Curve

- (a) It is “bell-shaped” with the peak occurring at $x = \mu$.
- (b) It is symmetric about the vertical line $x = \mu$.
- (c) It is concave down in the range $\mu - \sigma \leq x \leq \mu + \sigma$.
- (d) It is concave up outside that range, with inflection points at $x = \mu - \sigma$ and $x = \mu + \sigma$.

Figure 2: Properties of the normal distribution

Consider the function $g(X_n)=cn$. The output for this function are the cost
 85 values c calculated from the input random sequences in X_n , with n finite and
 positive. The sequences are the semantic valid programs encoded by Bernoulli
 strings using a Shannon-Fano code. A Turing Machine executing these programs
 must decide if to halt by interpreting the random signals read from the tape. If
 the signal is statistically significant and the entropy is reduced then the program
 90 can stop and return the decision in binary representation. The computational
 complexity to decide the problem is additive to the machine implementing the
 program up to a limiting point.

As an example let p_1, p_2, p_3 be a given set of programs and let the costs
 for $g(X = 001 = p_1) = c_1, g(X = 111 = p_2) = c_2, g(X = 101 = p_3) = c_3$. Let
 95 the acceptance threshold T for the machine M returns False when $g(X)$ is off
 by more than 1%. Assuming that the cost values c_1, c_2, c_3 is a normal random
 variable with mean 50 and standard deviation 0.5. The range for the programs
 in this domain are $49.5 \leq g(X) \leq 50.5$. The probability that a program
 will be accepted is $P(49.5 \leq X \leq 50.5)$. Let $f=g(X)$ be a normal random
 100 variable with median $m= 50$ and standard deviation $SD = 0.5$. Therefore from
 [6] we can see that 68.27% of the programs will be accepted and the remaining
 31.73% will be rejected. The calculations are shown in figure 3:

$$P(49.5 \leq X \leq 50.5) = \int_{49.5}^{50.5} f(x) dx,$$

$$\begin{aligned} f(x) &= \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \\ &= \frac{1}{0.5\sqrt{2\pi}} e^{-\frac{(x-50)^2}{0.5}}. \end{aligned}$$

$$P(49.5 \leq X \leq 50.5) = \int_{49.5}^{50.5} f(x) dx \approx .6827.$$

Figure 3: Probability for $P(49.5 \leq X \leq 50.5)$

The normal distribution for $Y=g(X)$ with mean = 50 and SD = 0.5 is shown in figure 4:

Figure 4: The normal distribution of $P(49.5 \leq X \leq 50.5)$ for $g(X)$ with mean = 0.5 and SD = 0.5

105 3.1.2. Normality test

To determine if a data-set can be modeled as a normal distribution we can use the Shapiro-Wilk method to test the null hypothesis that the sample means came from a normal distributed population. In figure 5 its defined as:

$$W = \frac{\left(\sum_{i=1}^n a_i x_{(i)}\right)^2}{\sum_{i=1}^n (x_i - \bar{x})^2},$$

Figure 5: Shapiro-Wilk formula

As an example, consider the cost samples from $g(X) = \{50, 51, 49, 49.3, 50.5, 50.3, 49.1, 48\}$ and a significance level of $\alpha=0.05$. The sample size, average and standard deviation are show in the figure 6:

Sample size (n):	8
Average ($\bar{x}$):	49.650000
Median:	49.65
Sample Standard Deviation (S):	0.975412
Sum of Squares:	6.660000
b:	2.538630
Skewness:	-0.325111
Skewness Shape:	▲ Potentially Symmetrical (pval=1.334)
Excess kurtosis:	-0.455645
Tails Shape:	▲ Potentially Mesokurtic, normal like tails (pval=1.242)
P-value:	0.950758
Outliers:	

Figure 6: The data distribution and p-value from the Shapiro-Wilk test for the function $g(X)$.

Since the $p\text{-value} > \alpha$ we accepted the H_0 and we can assume the data is normally distributed. The distribution and the histogram are reproduced in figure 7 and 8:

Figure 7: The normal distribution for function $g(X)$.

Figure 8: The probability density function for function $g(X)$.

115 3.1.3. Chi-square test

The Chi-square test can be used to determine whether there is a significant difference between expected and observed frequencies in one or more categories. The observations are classified into mutually exclusive classes according to a

hypothesis described by the probability of observation of a value in each cor-
 120 responding class. The 2x2 contingency table for the chi-square can be used to
 compare two groups of dichotomous dependent variables. The formula for the
 Chi-Square test is defined in figure 9:

$$X^2 = \sum_{i=1}^r \sum_{j=1}^c \frac{(O_{i,j} - E_{i,j})^2}{E_{i,j}}.$$

Figure 9: Chi-square test formula

As an example consider two programs pA and pB that encodes 600 Bernoulli
 strings. Each program has a set of Bernoulli sequences that were classified
 125 between two groups: 0 and 1. Group 0 is the set with sequences with better
 costs than a given threshold (i.e. improves expected quality) and the group 1
 is the sample-set with sequences with worse or equal quality (i.e. reduces or
 does not change the expected quality of $g(X)$). A chi-square test can be used
 to determine whether there is a significant difference between the proportion of
 130 sequences in programs pA and pB. The Contingency Table is show in figure 10.

	0 (Quality Improvement)	1 (Quality Reduction)	Marginal Row Totals
Process A - pA	139 (155) [1.65]	161 (145) [1.77]	300
Process B - pB	171 (155) [1.65]	129 (145) [1.77]	300
Marginal Column Totals	310	290	600 (Grand Total)

Figure 10: Contingency Table for the expected value of $g(X)$ relative to random variable X.

The chi-square statistics is 6.8343. The p-value is .008943. There is enough
 evidence to reject the null hypothesis as p-value is less than $\alpha = 0.05$.

3.2. Information Theory

3.2.1. Entropy of a random sequence

135 Entropy is a measure of uncertainty of a random variable. In other words it's the average rate at which information is produced by a stochastic process. Shannon defined the entropy H as a discrete random variable X with possible values as outcomes draw from a probability density function $P(X)$ [1]. In figure 11 the entropy function is defined as:

$$H(X) = - \sum_{i=1}^n P(x_i) \log_b P(x_i)$$

Figure 11: Shannon Entropy formula

140 The conditional entropy of two events $X = x$ and $Y = y$ with $X = x_i$ and $Y = y_j$ for (i,j) and $p(x_i, y_j)$ is the amount of randomness in the random variable X given Y . In figure 12 this relationship is defined as:

$$H(X|Y) = - \sum_{i,j} p(x_i, y_j) \log \frac{p(x_i, y_j)}{p(y_j)}$$

Figure 12: Conditional entropy

Let consider for example a binary program $p1$ with probability $p=0.7$ for halting and returning True (state A) and $1-q$ for returning False (state B). At
145 every trial one state is more likely to come up than the other. The reduction in entropy is expressed in a lower entropy value. The entropy for $p \neq q$ with $p=0.7$ is demonstrated in figure 13 :

$$\begin{aligned}
H(X) &= -p \log_2(p) - q \log_2(q) \\
&= -0.7 \log_2(0.7) - 0.3 \log_2(0.3) \\
&\approx -0.7 \cdot (-0.515) - 0.3 \cdot (-1.737) \\
&= 0.8816 < 1
\end{aligned}$$

Figure 13: Entropy for p=0.7

3.2.2. Bernoulli distribution and the probability of success

If a random process can only have one of two possible outcomes (Success (True) and Failure (False)) with probabilities p and $1-q$ respectively than we have a Bernoulli trial. Let X be a random variable with $X=1$ for the "success" state outcome and $X=0$ for the "failure" state. The indexing function for event $\{X = 1\}$ is defined as $I\{X_k = 1\} = 1$ if $X=1$ and $I\{X_k = 1\} = 0$ if $X=0$ than X is a random Bernoulli variable with probability distribution defined as:

155 $p_x(X_k) = p \times I\{X = 1\} + q \times [1 - I\{X = 1\}]$ and $p_x(X_k) = 0$ otherwise. The median (expected value) and the variance $\text{Var}[X]$ of the Bernoulli variable are defined by $E(X) = p$ and the $\text{Var}[X] = pq = p(1-p)$.

Consider for example a Bernoulli trial with size n for random variable X . Let $g(x)$ be the utility function for X and p be the probability of a program p_n producing as output a string which optimizes $g(X)$. Let $K_c(X)$ be the length of the shortest program that outputs X and halts. Let $H(X)$ and $H(g(X))$ be the entropy function for variable X and the function $g(X)$ respectively. The only possible results are the event of interest F (i.e. finding the optimal solution to any decision problem and thus maximizing the utility function) and event

165 $!F$ (i.e. not finding the optimal solution) with probabilities $p=0.003$ and $q = 1-p = 0.997$. The probability function for this Bernoulli trial is $p_x(X_k) = (0.003) \times I\{X = 1\} + 0.997 \times [1 - I\{X = 1\}]$ and 0 otherwise. For event F (Success state) let $I[X=1]$ be the indexing function for 1 value and 0 when the outcome is event $!F$ (Failure state). The median m and variance are $m=0.003$

170 and $\text{Var}[X] = (0.003)(0.997) = 0.003$.

3.2.3. Random variables and utility function

Let X be an independent random variable with alphabet $L : \{001, 010, 100, \dots\}$. The utility function $Y=g(X)$ of a random variable X express the preference of a given order of possible values of X . This order can be a logical evaluation of the value against a given threshold or constant. As an example if $g(X1 = 001) = c1$ and $g(X2 = 010) = c2$ are the costs of two routes between a set of nodes - in a super-graph G^* - we can use this function to determine the arithmetical relationship between them and decide if $c1$ is less, greater or equal to $c2$. The probability density function $\text{pdf}(Y)$ can be used to calculate the entropy of the distribution of the cost values.

3.2.4. Entropy of the utility function

Let $g(X)$ be a utility function of a random independent variable X with entropy $H(g(X))$. We need to determine the condition in which the entropy of function $g(X)$ is not greater or equal than the entropy of X . This relationship is defined by $H(g(X)) \leq H(X)$. From Wiegand and Schwarz [7] we have the proof of this condition:

Using the chain rule, $H(X, Y) = H(X) + H(Y|X)$, we can write

$$\begin{aligned} H(X, g(X)) &= H(g(X), X) \\ H(X) + H(g(X)|X) &= H(g(X)) + H(X|g(X)) \\ H(g(X)) &= H(X) + H(g(X)|X) - H(X|g(X)). \end{aligned}$$

Since the random variable $g(X)$ is a function of the random variable X , the value of $g(X)$ is known if the value of X is known. Hence, the conditional probability mass function $p_{g(X)|X}(y|x)$ is given by

$$p_{g(X)|X}(y|x) = \begin{cases} 1 & : y = g(x) \\ 0 & : y \neq g(x) \end{cases}.$$

Let $\mathcal{A}_X$ denote the alphabet of the random variables X with $\forall x \in \mathcal{A}_X, p_X(x) > 0$. Similarly, let $\mathcal{A}_Y$ denote the alphabet of the random variable $Y = g(X)$ with $\forall y \in \mathcal{A}_Y, p_Y(y) > 0$. The conditional entropy $H(g(X)|X)$ is given by

$$\begin{aligned} H(g(X)|X) &= - \sum_{x \in \mathcal{A}_X} \sum_{y \in \mathcal{A}_Y} p_{X,g(X)}(x, y) \log_2 p_{g(X)|X}(y|x) \\ &= - \sum_{x \in \mathcal{A}_X} p_X(x) \left(\sum_{y \in \mathcal{A}_Y} p_{g(X)|X}(y|x) \log_2 p_{g(X)|X}(y|x) \right). \end{aligned}$$

The terms with $p_{g(X)|X}(y|x) = 0$ do not contribute to the sum in parenthesis and can be ignored. We obtain,

$$\begin{aligned} H(g(X)|X) &= - \sum_{x \in \mathcal{A}_X} p_X(x) p_{g(X)|X}(g(x)|x) \log_2 p_{g(X)|X}(g(x)|x) \\ &= -(1 \cdot \log_2 1) \cdot \sum_{\forall x} p_X(x) = -\log_2 1 \\ &= 0. \end{aligned}$$

Hence, the conditional entropy $H(g(X)|X)$ is always equal to 0, and we obtain for $H(g(X))$,

$$H(g(X)) = H(X) - H(X|g(X)).$$

Since the entropy is always greater than or equal to 0, we have proved that

$$H(g(X)) \leq H(X).$$

If and only if $g(X)$ is an injective function for all letters of the alphabet $\mathcal{A}_X$, i.e., if $\forall a, b \in \mathcal{A}_X, a \neq b$ implies $g(a) \neq g(b)$, we can define an inverse function $h(y)$, so that $h(g(x)) = x, \forall x \in \mathcal{A}_X$. In this case, we obtain

$$H(X|g(X)) = H(h(g(X))|g(X)) = 0,$$

Consequently, if $g(X)$ is an injective function for all letters of the alphabet $\mathcal{A}_Y$, the entropy $H(g(X))$ is equal to the entropy $H(X)$,

$$H(g(X)) = H(X).$$

If $g(X)$ is not an injective function for all letters of the alphabet $\mathcal{A}_Y$, i.e., if there are two alphabet letters a and $b \neq a$, with $g(a) = g(b)$, the entropy $H(g(X))$ is less than the entropy $H(X)$,

$$H(g(X)) < H(X).$$

3.2.5. Kolmogorov Complexity

The Kolmogorov complexity of a string w from language L denoted by $K_L(w)$ is the shortest program from alphabet L which produces w as output and halts.

[4] The conditional Kolmogorov complexity of string x relative to word w is defined by $K_L(w|x)$ and is the length of the shortest program that receives x as input and produces w as output.

3.2.6. Complexity of a string and shortest description length

Let U be a Universal computer. The Kolmogorov complexity $K_c(x)$ of a string x of a computer U is

$$K_c(x) = \minlength(p)$$

when

$$p : U(p) = x$$

It is the minimum length program p that output variable x and halts. Its the small possible program. Let C be another computer. If this complexity is

general there is a universal computer U that simulates C for any string x by a
 200 constant c on computer C . Thus $K_c(x)_U \leq K_c(x)_C + c$. This constant does
 not depend on string x . The upper bound on the Kolmogorov complexity is

$$K_c(x) \leq K_c(x|length(x)) + 2 \log_2 length(x) + c$$

Therefore there are few sequences with low complexity in the set. The lower
 bound states that are not many short programs in the set. If X is a Bernoulli
 sequence with probability $p(x) = 1/2$, this means there are no more than 2^k
 210 strings with complexity $K_c(x) < k$.

Consider each program can produce only one possible output, the number of
 sequence with complexity $K_c(x) < k$ is less than 2^k . Thus the complexity will
 depend on the computer up to an additive constant. Lets consider fractals for
 example. They can produce complex 2d images with self-replicating patterns
 210 with many different scales. However the complexity to describe its rules have a
 Kolmogorov Complexity close to zero. [2]

The expected value of the Kolmogorov complexity of a random sequence is
 close to the Shannon entropy. This relationship between complexity and entropy
 can be described as a stochastic process drawn to a i.i.d on variable X following
 215 a probability mass function $p(x)$. The symbol x in variable X is defined by an
 finite alphabet. This expectation is

$$E(1/n)K_c(X^n|n) \Rightarrow H(X)$$

Therefore most of the sequences in the set do not have simple description. How-
 ever there are some simple sequences.

The probability that a random sequence can be compressed by more than k
 220 bits is no greater than $2^{(-k)}$. Thus most Bernoulli sequences have a complexity
 close to their length.

The Kolmogorov complexity of a independent and identically distributed
 random binary variable created by a Bernoulli process is close to the entropy
 binary function. The expected value of the complexity of the Bernoulli sequence
 225 converges to the entropy. Therefore we can bound the number of sequences with

complexity that are significantly lower than entropy and classify each element between two sets (the typical sets and a non-typical set).

3.2.7. Kelly criterion and the uncertainty in random outcomes

Let $K(E(Y), H(Y)) = f$ be the maximization of the expected value of the
 230 logarithm of the entropy of the utility function $Y=g(X)$ [8]. This fraction is
 known as the Kelly criterion and can be understood as the level of uncertainty
 about a given data distribution of the random variable X relative to a probability
 density function $\text{pdf}(Y)$ of a measured respective cost distribution found at the
 sample. The value of f is a fraction of the cost-value of $g(X)$ on an outcome that
 235 occurs with probability p and odds b . Let the probability of finding a value
 which improves $g(X)$ be p and in this case the resulting improvement is equal
 to 1 cost-unit plus the fraction f . The probability of decreasing quality for Y is
 1- p . Therefore the expected value for log cost (E) is differed in figure 14 as:

$$E = p \log(1 + fb) + (1 - p) \log(1 - f)$$

Figure 14: Expected value of the utility function

From "Kelly criterion" in Wikipedia [5], "To find the value of f for which
 240 the expectation value is maximized, denoted as f^* , we differentiate the above
 expression and set this equal to zero.". In figure 15 we have the maximization
 of the expected value.

$$\frac{dE}{df^*} = \frac{pb}{1 + f^*b} - \frac{1 - p}{1 - f^*} = 0$$

Rearranging this equation to solve for the value of f^* gives the Kelly criterion:

$$f^* = \frac{pb + p - 1}{b}$$

Figure 15: Maximization of $E(X)$

Where f^* is the optimal fraction, b is the net odds, p is the probability of improving quality in $Y=g(X)$ and the q is the probability of decreasing quality
 245 $(1-q)$.

For example consider a program with a 60% chance of improving the utility function $g(X)$ thus $p=0.6$ and $q=0.4$. The program has a 1-to-1 odds of finding a sequence which improves $g(X)$ and thus $b=1$. For this parameters the program has a 20% ($f^*=0.20$) of confidence that the outcomes produce values that improve
 250 the expected value of $g(X)$ over many trials.

3.3. General Decision Machine

From Hopcroft & Ullman [9] a single-tape Turing Machine can be defined as a 7-Tuple $TM = \langle Q, \Gamma, b, \Sigma, \delta, q_0, F \rangle$ where Q is a finite set of states, Γ is a finite tape with symbols from a given alphabet L_0 , b is the blank symbol
 255 and $b \in \Gamma$, $\Sigma \subseteq \Gamma - \{b\}$ is the set of input symbols, $q_0 \in Q$ is the initial state, $F \subseteq Q$ is the set of final accepted states, $\delta : Q - F \times \Gamma \rightarrow Q - F \times \Gamma \times L, R$ is a partial function called the transition function where L and R are the Left and Right shift respectively.

Let X be a random symbol from the alphabet L_0 . We can expand the
 260 standard Turing Machine model to a general computation under a given know distribution with probability p . Consider the utility function $g(X)$ related the the machine TM. The output of $g(X)$ is dependent on the changes in the value of x in X . Assume that the values of $g(x)$ follow a normal distribution. Let the probability density function of $g(X)$ be $\text{pdf}(g(X))$. The entropy H from $g(X)$
 265 relative to X can be calculated. Let $K(E(Y), H(Y)) = f^*$ be the maximization of the expected value of the logarithm of the entropy of the utility function $Y=g(X)$.

3.4. Indexing binary function

Let the $C_p(X, g(X))$ be the p-value from the Chi-Square Test for random
 270 samples. Assume the samples are normally distributed. These functions evaluate whether the sample means are generated from the same distribution. Each

element of the sample is a random independent variable X that is compared to the output of the utility function $Y=g(X)$. This utility function is a conditional statement that returns(i.e. produces as a output) True or False by evaluating
 275 if a given x in $g(X)$ is greater, small or equal to a given threshold defined as a real value T . Let $E(Y)$ be the expected value of Y with entropy $H(Y)$. The function $K(E(Y), H(Y))$ is the uncertainty level about the state distribution between the random samples. In other words, this value represents the error rate in the representation of information in a give population under some de-
 280 grees of freedom. The function K is the maximization of the expected value of the logarithm of the utility function $g(X)$ of a random variable X .

The domain for function $C_p(X, g(X))$ is the range from 0 (no change) to 1 (high confidence). The value outputted by this function is the significance level. For example a value of 0.05 represents a 5% chance. The domain for function
 285 K is $(-\infty, +\infty)$. The K value represents the intensity about the uncertainty of randomly finding state outcomes that improves (or reduces) the utility function $Y=g(X)$. If the K value is positive then it shows the amount of useful information encoded in the sample. For example if the value is 0.05 (5%) then the uncertainty is reduced (i.e decrease in entropy) and there is additional side information
 290 transmitted by the information source. If the edge is negative the K value is also negative indicating that the chance of finding useful information from the elements in the random sample is unlikely.

Let M be an index function that classifies a sample according to the distribution of elements in probability distribution of a function $g(X)$. The classification
 295 labels each element in the sample between Accept/True or Reject/False. After the categorization of all input values the list of executions can be sorted by this label. In other words M is the sort key in a rank of sequences. The function M is defined by $M = C_p(X, g(X))/K(E(Y), H(Y))$

where X is a Bernoulli sequence, $Y=g(X)$ is the utility function, $H(Y)$ is the
 300 entropy for Y , K is the uncertainty level, $E(Y)$ the expected value of Y . The domain is $(-\infty, +\infty)$.

Let's consider for example two groups (alpha and beta) of random Bernoulli

sequences for a trial of size 100. Both samples are assumed to be normal distributions. Let pA be a program that needs to decide to halt based on the
 305 expectation of the output values for the utility function $g(X)$. The fixed alpha sample is the known model-reference set and the beta sample is the alternative sample hypothesis. Both samples have 300 candidate solution sequences each ($N=600$). The computed executions with chi-square p-value less than 0.05 and Kelly fraction greater than 0 are the solutions of interest and they produce
 310 greater quality by reducing entropy at each interaction. Let's consider the subset s^* of sequences produced at interaction $s^* = \{35, 69, 4, 83, 41, 40\}$. We want to classify each element of s^* between two categorical classes: the near optimal sequence pools (interaction 35, 69 and 4) and another random sample with discarded sequence samples (83, 41, 40). In the figure below, we can see the
 315 p-values and Kelly fractions for instance 4, 35, 69 (good near optimal solutions) and 41, 40 and 83 (random bad solutions). We want to filter solution sequences with candidate solutions that produce significant results and have a maximum rate of decrease in entropy. In other words, there is less uncertainty about the possible favorable state outcomes - relative to $g(X)$ and $E(Y)$ - with the new
 320 alternative sample against the previous known best solution hypothesis set. In the figure 16 we have the representation of the probability density function, the p-value for the Chi-square test and the Kelly fraction relative to the measured level of uncertainty.

Beta sample size: 300									
Alpha sample size: 300									
Expected Cost in the Alpha sample: 354.08									
Iteration	0-Group in Alpha	1-Group in Alpha	0-Group in Beta	1-Group in Beta	Chi-square	p-value			
35	139	161	161	171	129	0.008942571850326772			
69	139	161	161	167	133	0.022216388713348016			
4	139	161	161	165	135	0.03374716230794477			
83	139	161	161	156	144	0.16506488542811487			
41	139	161	161	155	145	0.19132960690416181			
40	139	161	161	151	149	0.32691797952947926			
Iteration	Pr[0-Group] in Alpha and Beta	Pr[1-Group] in Alpha and Beta	Average Quality Improvement	Average Quality Loss	Quality (Increase/Decrease)	Ratio	Kelly fraction		
35	0.5166666666666667	0.4833333333333334	43.44112696024912	41.04564146800691	1.058361507		0.05998591		
69	0.51	0.49	42.63546620761791	39.47314129784540	1.080113333		0.056343902		
4	0.5066666666666667	0.4933333333333335	43.10372527365191	39.93298331557616	1.07940158		0.049623299		
83	0.4916666666666664	0.5083333333333333	43.82162826721773	41.34019759738970	1.060024645		0.012118068		
41	0.49	0.51	44.19721729785729	41.82991802412611	1.056593448		0.007316711		
40	0.4833333333333334	0.5166666666666667	42.80351230003428	39.68076739847514	1.078696686		0.00436026		

Figure 16: Data distribution for the alpha and beta groups.

By comparing the p-value and the Kelly fraction any program p_n can decide
 325 whatever to halt based on “side information” on which symbols can lead - in the
 long run - to the best possible rate of quality improvement for the utility function
 $g(X)$, over finite many interactions. Therefore the halting problem is reduced
 to a sort problem using the p-value and the Kelly fraction as the primary and
 secondary keys for an unsorted computation list. To find the best near optimal
 330 sequence any program can sort the keys by decreasing order. Additionally it
 can sort the sub-list by the number of bits required to encode the binary -
 near optimal - string in a Shannon-Fano code following the probability density
 function $\text{pdf}(g(X))$. The relationship between the p-value and the kelly fraction
 is show in figure 17:

Figure 17: Bar chart for the p-value from the Chi-square test and the Kelly fraction

3.5. Applications

The indexing binary function M can be used as a key to decide if a given
 decision problem should halt or not. For example the Traveling Salesman Prob-
 lem (TSP) wants to find the shortest route in a graph of connected nodes. The
 function M can label each output and decide whether to halt if the intensity of
 340 M is above a given threshold T . Another example is the game of Sudoku where
 the candidate solution that has a better partial solution has a better score than

other alternatives. In other words the state of any Bernoulli sequence has a proportional real value that ranges according to the variation in entropy of the utility function of the candidate strings. Both Sudoku and the TSP are in the
345 class of non-polynomial (NP) class of problems [10].

In this paper we have proposed a new approach for risk/uncertainty modeling based on the Kelly-Shannon-Thorp criterion to measure the optimal channel allocation that a program should place at each interaction of an algorithm (implemented in a Turing Machine) for solving any non polynomial decision problem
350 such as the Traveling Salesman Problem. This new interpretation of the NP problems by means of Information Theory and Kolmogorov-Chaitin Complexity can be used to define the limits of computation and answer the question of which cases $P=NP$. It works by setting the boundaries of the problem as a probability density function draw according to a sample of random solution
355 samples defined as semantic valid Bernoulli binary strings. Each candidate solution cost is evaluated against a binary statistical indexing function M in order to classify it between 2 independent groups: 0 (smaller cost — better quality); 1 (greater or equal cost — worse or unchanged quality). This distribution reveals the amount of inside information available to the program and the entropy can
360 be calculated. The Turing Machine can then use this information to reduce entropy at each execution clock unit while maximizing the expected improvement in solution quality (ie finding a shortest euclidean distance tour that visits all nodes as described in the TSP problem). In the TSP, the salesman receives a list of cities and paths from a “wire” or “tape” and must decide between accepting
365 any given new route against the current know best route. If he is lucky and chooses the right path, he may very well end with a smaller distance to travel across all cities and returning home. If he chooses the wrong path he may end up with thinking the current total distance is the best one even though it’s not. For the classical salesman there is no way to know how good or bad his path is
370 because he has no inside information about the overall expected distance of the routes. At each execution clock he must decide using only his current knowledge for that iteration about the best distance so far and the new alternative

solution.

Alternatively, the log-normal salesman's can improve his strategy in the long
 375 run by quantifying the total of available inside information in the channel (or a
 tape in the Turing machine) and maximizing the expected value of the logarithm
 of value function (defined by Traveled Euclidean distance) for each execution
 clock. Using this approach, he can reduce his uncertainty while optimizing his
 rate of distance reduction (quality improvement) at each execution time. He can
 380 do it by creating the probability density function of the cost distance variable
 by sampling two sets (current and new) of random tours distances.

3.5.1. Algorithmic-Information Language

Let L be a finite language with N symbols in a set such as $L : \{NODE_A; NODE_B; NODE_C, \dots NODE_N\}$.
 Each item is an identification label for the elements in the set under some prob-
 385 ability distribution function $pdf(x)$. This is the likelihood of choosing a given
 symbol x in L . This can be interpreted as a random variable x in L that can
 assume any item in the set by following the known distribution $pdf(x)$. Each
 symbol represents a city with a 1-1 mapping to a respective node in a super-
 graph G^* . Therefore, each node is a point with X_n and Y_n coordinates in a
 390 2D Euclidean space. Let the function $g(X)$ be the utility function for a given
 language L . This functions outputs the costs or weight of X relative to a given
 threshold T defined as the expected value of $E(g(X))$. Assume $g(X)$ is a Bernoulli
 distribution.

For example consider the alphabet L that describes 3 cities encoded by 3
 395 symbols in the set $L : \{A(x1, y1), B(x2, y2), C(x3, y3)\}$. Thus, each element is a
 node that have two variables that stores the x and y axis coordinate accordingly.
 A valid sequence s^* can be produced by combining elements of L . Permutations
 with duplicate elements are invalid. S^* is the set with all possible permutations
 of this sequence. The neighbor of each node at position i in the sequence is
 400 the next connected city (i.e item at position $i+1$). This sequence describes a
 path between nodes and is a tour starting from the first node until the last and
 returning to the first node when finished [11]. This is shown in figures 18 and

19:

A sample graph for the traveling salesman problem and the corresponding cost matrix.

Figure 18: Cost Matrix

In figure 19 we have an example of valid and invalid tours sequences:

Word in Language	Path Sequence	Total cost/weight distance
Valid	A->B->C->D->E->A	Sum of total Euclidian Cost 1
Valid	D->B->C->A->E->D	Sum of total Euclidian Cost 2
Valid	D->B->A->C->E->D	Sum of total Euclidian Cost 3
Invalid	A->B->A->D->E->A	Sum of total Euclidian Cost 4

Figure 19: Sample path strings

405 In the figure 20 and 21 we can see an example of the probability density function for two random samples with N=4 for both sets: Group A and B. The new expected cost for the sample Group B is 49.25 units . For example if the expected cost for the current sample was 50 (in Group A) there are 2 solutions in B and 1 candidate in A which improve the solution quality and reduces the

410 traveling distance. The probability of finding a candidate solution at random in group A that reduces cost is 25%. The probability of finding a candidate solution at random in group B that reduces cost relative to the expected cost distance from A is 50%.

Current path route (A)	Current total distance cost A	Small than current expected cost A	Distance difference
ABCDEA	50	False	0
ACBDEA	52	False	+2
ACDBEA	48	True	-2
AEBDCA	50	False	0

Figure 20: Group A candidate solution sample

415 The average quality improvement from B is 26.5 units with distance reduction from expected cost of 50 to 35 and 50 to 12. The average quality decrease or unchanged is 25 units, from 50 to 50 and 50 to 100. The rate of improvement is the average distance reduction divided by the average quality reduction. For this example, $26.5/25 = 1.06$. In the figure 21 we can see the data distribution for group B relative to the costs from E(A):

New path route (B)	New total distance cost B	Small than current expected cost A	Distance difference
ABCDEA	50	False	0
ACBDEA	35	True	-15
BACDEB	12	True	-38
CEBADC	100	False	+50

Figure 21: Group B candidate solution sample

420 The Kelly criterion is 0.0283 (2.83%). This can be interpreted as a measurement of risk that a salesman has on choosing the wrong path (with larger distances). In simple terms, it's the level of uncertainty about the overall possible states. This is the error rate of transmission of information in a noisy communication channel. If the fraction is negative the salesman must switch

425 strategy and stay with current know best path sample and wait for the next
alternative sample that will be created at the next execution iteration clock.
[12].

Solution Group B relative to Expected Cost E(A)		
Average Solution Cost		Average Absolute Cost Reduction
	49,25	26,5
Average Absolute Cost Increase		Rate of Quality Improvement
	25	1,06
Probability Cost Increase		Probability of Cost Decrease
	0,5	0,5
Kelly fraction	0,028301887	

Figure 22: Kelly fraction to measure information uncertainty

The entropy of the new cost output is one full bit of information and is show
in figure 23

$$\begin{aligned} H(X) &= - \sum_{i=1}^n P(x_i) \log_b P(x_i) \\ &= - \sum_{i=1}^2 \frac{1}{2} \log_2 \frac{1}{2} \\ &= - \sum_{i=1}^2 \frac{1}{2} \cdot (-1) = 1 \end{aligned}$$

Figure 23: Entropy for p =0.5 and q=0.5 with p = 1 — q

430 The contingency table for sample is demonstrated in figure 24:

	Reduce Distance	Increase Distance	Marginal Row Totals
Group A	1	3	4
Group B	2	2	4
Marginal Column Totals	3	5	8 (Grand Total)

Figure 24: Contingency Table for current (Group A) and alternative (Group B) solution candidates relative to expected current cost from Group A.

Consider for example a larger solution sample. The current solution group (A) and the alternative solution group (B). Group A has 100 solution sequences and B also has 100 candidates. There are 25 possible candidate tours that reduce the distance in A. In group B there are 50 candidates that increase the average cost relative to current expected distance from A. The salesman can use the chi-square test statistic to determine if there is a significant difference between the sample means. This can be used to determine if the expected distance decrease from the candidate solutions in group A (i.e. null hypothesis) is equivalent to the performance of the alternative candidates in group B (i.e. alternative hypothesis). The chi-square statistic and p-value are shown in the figure 25

Results					
	Reduce Distance	Increase Distance			Row Totals
Group A	25 (37.50) [4.17]	75 (62.50) [2.50]			100
Group B	50 (37.50) [4.17]	50 (62.50) [2.50]			100
Column Totals	75	125			200 (Grand Total)

Figure 25: Contingency Table for group A and B relative to E(A)

The chi-square statistic is 13.3333. The p-value is .000261. The result is significant at $p < .05$. With a significance level of 0.05 there is strong evidence against the null hypothesis and there is a 5% chance the null is correct. We can reject the null hypothesis and accept the alternative hypothesis that Group B has

solutions that produce better quality than in Group A.

4. Conclusion

In this paper we have demonstrated the statistical analysis of a stochastic process created by a set of Shannon-Bernoulli sequences. From this framework
 450 it's possible to analyze the distribution of information for a given random variable and the correlation between this variable and a utility function that compares the differences between state sequences. This function shares information with the random variable itself and this side information can be measured. The knowledge of this side information can be used by any program - executing in
 455 a Turing Machine - to decide whatever to halt or not.

The advantage of this approach is that it is independent of the computer encoding the problem and the time and space complexities are additive up to a limit that is linearly proportional to the input size. Other heuristics methods assume a predefined knowledge about the data structure and thus are biased
 460 towards this encoded knowledge. The implications of this interpretation is that by reducing the NP problems to a matter of modularization of encoded and decoded random signals in an alphabet we can find near optimal solutions that are statistically significant and are guaranteed to produce the best rate of improvement in the long run over many trials(i.e. simulation iterations) even though
 465 the problems itself is computationally complex and the alternative sequential brute force algorithm would require exponential time to solve.

The traditional interpretation of the halting problems says there is no general algorithm to solve the decision problem for all possible program-input pairs. However, as demonstrated in this paper, the decision problem itself can be quantified in bits and the distribution of random sequences that are - syntactically
 470 and semantically - valid is not neglectable. In other words, although not all sequences produce outputs that optimizes the quality for any utility function $g(X)$, there are some infrequent random sequences that improve $g(X)$ and are statistically significant under some given degree of freedom. This new inter-

475 pretation of information rate can be used to define the limitations imposed by
the computational complexity of Turing Machines when solving non polynomial
problems, regardless of time and space constraints.

References

- [1] C. E. Shannon, A mathematical theory of communication., 27:379 – 423,623
480 – 656,, Bell Syst. Tech.J.
- [2] T. M. Cover, Elements of information theory, John Wiley Sons, 2012.
- [3] na., Kolmogorov complexity, Wikipedia.
URL https://en.wikipedia.org/wiki/Kolmogorov_complexity
- [4] A. N. Kolmogorov, Logical basis for information theory and probability
485 theory, IEEE Trans. Inf. Theory.
- [5] Wikipedia, Kelly criterion, Wikipedia.
URL https://en.wikipedia.org/wiki/Kelly_criterion
- [6] na, Calculus applied to probability and statistics., na.
URL [https://www.cengage.com/resource_uploads/downloads/](https://www.cengage.com/resource_uploads/downloads/1439049254_242719.pdf)
490 [1439049254_242719.pdf](https://www.cengage.com/resource_uploads/downloads/1439049254_242719.pdf)
- [7] T. Wiegand, H. Schaar, Berlin institute of technology. source coding., na.
URL [https://www.ic.tu-berlin.de/fileadmin/fg121/](https://www.ic.tu-berlin.de/fileadmin/fg121/Source-Coding_WS12/solutions/solutions_part_01.pdf)
Source-Coding_WS12/solutions/solutions_part_01.pdf
- [8] J. K. 1956, A new interpretation of information rate, J,35:917–926, July .
495 Bell Syst. Tech. (1956).
- [9] J. E. Hopcroft, J. D. . Ullman, Formal languages and their relation to au-
tomata, Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA.
- [10] na, Traveling salesman problem, Wikipedia.
URL https://en.wikipedia.org/wiki/Travelling_salesman_problem

- 500 [11] M. S. Lima, Algorithmic-information theory interpretation to the traveling salesman problem., Preprintdoi:<https://doi.org/10.31224/osf.io/gmzn5>.
- [12] M. S. Lima, The log-normal messenger problem. an information theory solution to the traveling salesman problem, Medium.
- 505 URL <https://medium.com/@santanalima.matheus/the-log-normal-messenger-problem-c4a0257351ee>