Kolmogorov Complexity and Information Theory

with an interpretation in terms of questions and answers

Peter D. Grünwald* and Paul M.B. Vitányi†
CWI, P.O. Box 94079, NL-1090 GB Amsterdam, the Netherlands
E-mail: {pdg,paulv}@cwi.nl

Abstract. We compare the elementary theories of Shannon information and Kolmogorov complexity, the extent to which they have a common purpose, and where they are fundamentally different. We discuss and relate the basic notions of both theories: *Shannon entropy, Kolmogorov complexity, Shannon mutual information* and *Kolmogorov ('algorithmic') mutual information.* We explain how *universal coding* may be viewed as a middle ground between the two theories. We consider Shannon's rate distortion theory, which quantifies *useful* (in a certain sense) information. We use the communication of information as our guiding motif, and we explain how it relates to sequential question-answer sessions.

Keywords: Kolmogorov complexity, algorithmic information theory, Shannon information theory, mutual information, prefix codes, universal codes, rate distortion theory, data compression

1. Introduction

How should we measure the amount of information about a phenomenon that is given to us by a particular observation concerning the phenomenon?

Shannon information theory, usually called just 'information' theory, was introduced in 1948 by C.E. Shannon (1916–2001). *Kolmogorov complexity* theory is also known as 'algorithmic information' theory. It was introduced independently and with different motivations by R.J. Solomonoff (born 1926), A.N. Kolmogorov (1903–1987) and G. Chaitin (born 1943) in 1960/1964, 1965 and 1966 respectively. Both theories aim at providing a means for measuring 'information'. They use the same unit to do this: the *bit.* In both cases, the amount of information in an object may be interpreted as the length of a description of the object. In the Shannon approach, however, the method of encoding objects is based on the presupposition that the objects to be encoded are outcomes of a known random source—it is only the characteristics of that random source that determine the encoding, not the characteristics

* Supported in part by a travel grant awarded by the Netherlands Organization for Scientific Research (NWO).

† Supported in part by the EU project RESQ, IST-2001-37559, the NoE QIPROCONE +IST-1999-29064 and the ESF QiT Programme.

of the objects that are its outcomes. In the Kolmogorov complexity approach we consider the individual objects themselves, in isolation so-to-speak, and the encoding of an object is a computer program (Turing machine) that generates it and then halts. In the Shannon approach we are interested in the minimum expected number of bits to transmit a message from a random source of known characteristics through an error-free channel. In Kolmogorov complexity we are interested in the minimum number of bits from which a particular message can effectively be reconstructed. A little reflection reveals that this is a great difference: for *every* source emitting but two messages the Shannon information is at most 1 bit, but we can choose both messages concerned of arbitrarily high Kolmogorov complexity. Shannon stresses in his founding article that his notion is only concerned with *communication*, while Kolmogorov stresses in his founding article that his notion aims at supplementing the gap left by Shannon theory concerning the information in individual objects. To be sure, both notions are natural: Shannon ignores the object itself but considers only the characteristics of the random source of which the object is one of the possible outcomes, while Kolmogorov considers only the object itself to determine the number of bits in the ultimate compressed version irrespective of the manner in which the object arose.

How to read this paper In this paper, we introduce, compare and contrast the Shannon and Kolmogorov approaches. We do this by switching back and forth between the two theories, according to the following pattern: we first discuss a concept of Shannon's theory, discuss its properties as well as some questions it leaves open. We then provide Kolmogorov's analogue of the concept and show how it answers the question left open by Shannon's theory. We use as our guiding motif the communication between a sender A and a receiver B; where appropriate, we also discuss the related setting of a question-answer session between B and A.

To obtain an understanding of the two theories and how they relate, it is crucial to read the overview below and then Sections 2 and Section 3, which discuss preliminaries, fix notation and introduce the basic notions. The other sections are written in a way so that they can be read separately from one another. Throughout the text, we assume some basic familiarity with elementary notions of probability theory and computation.

The paper does not contain any new results. All theorems that we present here, as well as further details, context and discussion, can be found in either of two standard text books: (Cover and Thomas, 1991), the standard reference on Shannon information theory, and/or (Li and Vitányi, 1997), the standard reference on Kolmogorov complexity.

1.1. Overview and Summary

A summary of the basic ideas is given below. In the paper, these notions are discussed in the same order.

1. **Coding:**
   Prefix codes, Kraft inequality. Since descriptions or *encodings* of objects are fundamental to both theories, we first review some elementary facts about coding. The most important of these is the *Kraft inequality.* This inequality gives the fundamental relationship between *probability mass functions and prefix codes*, which are the type of codes we are interested in (Section 2).

2. **Shannon's Fundamental Concept:**
   Entropy is defined as a functional that maps *probability distributions* or, equivalently, *random variables*, to *real numbers.* This notion is derived from first principles as the only 'reasonable' way to measure the 'average amount of information conveyed when an outcome of the random variable is observed'. The notion is then related to encoding and communicating messages by Shannon's famous 'coding theorem' (Section 3.1).

3. **Kolmogorov's Fundamental Concept:**
   Kolmogorov Complexity is defined as a function that maps *objects* (to be thought of as natural numbers or sequences of symbols) to the *natural numbers*. Intuitively, the Kolmogorov complexity of a sequence is the length (in bits) of the shortest computer program that prints the sequence and then halts (Section 3.2).

4. **Universal Coding:**
   interpolating between Shannon and Kolmogorov. Although their primary aim is quite different, and they are functions defined on different spaces, there are close relations between entropy and Kolmogorov complexity. These are best illustrated by explaining 'universal coding' which combines elements from both Shannon's and Kolmogorov's theory, and which lies at the basis of most practical data compression methods (Section 4).

Entropy and Kolmogorov Complexity are the basic notions of the two theories. They serve as building blocks for all other important notions in the respective theories. Arguably the most important of these notions is *mutual information*:

5. Mutual Information for Shannon and Kolmogorov:
Entropy and Kolmogorov complexity are concerned with information in a single object: a random variable (Shannon) or an individual sequence (Kolmogorov). Both theories provide a (distinct) notion of *mutual information* that measures the information that *one object gives about another object.* In Shannon's theory, this is the information that one random variable carries about another; in Kolmogorov's theory ('algorithmic mutual information'), it is the information one sequence gives about another (Section 5).

Entropy, Kolmogorov complexity and mutual information are concerned with *lossless* description or compression: messages must be described in such a way that from the description, the original message can be completely reconstructed. Extending the theories to *lossy* description or compression enables the formalization of more sophisticated concepts, such as 'meaningful information' and 'useful information'. 'Meaningful information' is defined in the Kolmogorov framework using the *Kolmogorov structure function.* 'Useful information' is defined in Shannon's framework using the *rate-distortion function.* We end the paper with a brief treatment of the latter:

6. Useful Information:
Rate-distortion theory is the part of Shannon information theory that deals with the following situation: The sender is only allowed to use a fixed (small) number of bits to send his message. The goal is then to send the most *useful* or *valuable* message given this constraint (Section 6).

2. The Coding Framework and the Kraft Inequality

Notational preliminaries:
1. Strings Let $\mathcal{X}$ be some finite or countable set. We use the notation $\mathcal{X}^*$ to denote the set of finite *strings* or *sequences* over $\mathcal{X}$. For example,

$$\{0,1\}^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\},$$

with ϵ denoting the *empty word* '' with no letters. Let $x, y, z \in \mathcal{N}$, where $\mathcal{N}$ denotes the natural numbers. We identify $\mathcal{N}$ and $\{0,1\}^*$ according to the correspondence

$$(0,\epsilon), (1,0), (2,1), (3,00), (4,01), \ldots \tag{1}$$

The *length* $l(x)$ of x is the number of bits in the binary string x. For example, $l(010) = 3$ and $l(\epsilon) = 0$. If x is interpreted as an integer, we get $l(x) = \lfloor \log(x+1) \rfloor$ and, for $x \geq 2$,

$$\lfloor \log x \rfloor \leq l(x) \leq \lceil \log x \rceil. \tag{2}$$

Here, as in the sequel, $\lceil x \rceil$ is the smallest integer larger than or equal to x, $\lfloor x \rfloor$ is the largest integer smaller than or equal to x and log denotes logarithm to base two. We shall typically be concerned with encoding finite-length binary strings by other finite-length binary strings. The emphasis is on binary strings only for convenience; observations in any alphabet can be so encoded in a way that is 'theory neutral'.

2. (In)equality up to a constant We will denote by $\stackrel{+}{<}$ an inequality to within an additive constant. More precisely, let f, g be functions from $\{0,1\}^*$ to $\mathbf{R}$. Then by '$f(x) \stackrel{+}{<} g(x)$' we mean that there exists a c such that for all $x \in \{0,1\}^*$, $f(x) < g(x) + c$. We denote by $\stackrel{+}{=}$ the situation when both $\stackrel{+}{<}$ and $\stackrel{+}{>}$ hold.

3. Probabilities Let P be a probability distribution defined on a finite or countable set $\mathcal{X}$. Throughout this text, we denote by X the random variable that takes values in $\mathcal{X}$; thus $P(X = x) = P(\{x\})$ is the probability that the event $\{x\}$ obtains. We write both $P(x)$ and p_x as an abbreviation of $P(X = x)$.

Codes We repeatedly consider the following scenario: a *sender* (say, A) wants to communicate or transmit some information to a *receiver* (say, B). The information to be transmitted is an element from some set $\mathcal{X}$. It will be communicated by sending a binary string, called the *message*. When B receives the message, he can decode it again and (hopefully) reconstruct the element of $\mathcal{X}$ that was sent. To achieve this, A and B need to agree on a *code* or *description method* before communicating. Intuitively, this is a binary relation between *source words* and associated *code words*. The relation is fully characterized by the *decoding function*. Such a decoding function D can be any function $D : \{0,1\}^* \to \mathcal{X}$. The domain of D is the set of *code words* and the range of D is the set of *source words*. $D(y) = x$ is interpreted as "y is a code word for the source word x". The set of all code words for source word x is the set $D^{-1}(x) = \{y : D(y) = x\}$. Hence, $E = D^{-1}$ can be called the *encoding* substitution (E is not necessarily a function). With each code D we can associate a *length function* $L_D : \mathcal{X} \to \mathcal{N}$ such that, for each source word x, $L(x)$ is the length of the shortest encoding of x:

$$L_D(x) = \min\{l(y) : D(y) = x\}.$$

We denote by x^* the shortest y such that $D(y) = x$; if there is more than one such y, then x^* is defined to be the first such y in lexicographical order.

In coding theory attention is often restricted to the case where the source word set is finite, say $\mathcal{X} = \{1, 2, \ldots, N\}$. If there is a constant l_0 such that $l(y) = l_0$ for all code words y (equivalently, $L(x) = l_0$ for all source words x), then we call D a *fixed-length* code. It is easy to see that $l_0 \geq \log N$. For instance, in teletype transmissions the source has an alphabet of $N = 32$ letters, consisting of the 26 letters in the Latin alphabet plus 6 special characters. Hence, we need $l_0 = 5$ binary digits per source letter. In electronic computers we often use the fixed-length ASCII code with $l_0 = 8$.

Prefix code It is immediately clear that in general we cannot uniquely recover x and y from $E(xy)$. Let E be the identity mapping. Then we have $E(00)E(00) = 0000 = E(0)E(000)$. We now introduce *prefix codes*, which do not suffer from this defect. A binary string x is a *proper prefix* of a binary string y if we can write $y = xz$ for $z \neq \epsilon$. A set $\{x, y, \ldots\} \subseteq \{0,1\}^*$ is *prefix-free* if for any pair of distinct elements in the set neither is a proper prefix of the other. A function $D : \{0,1\}^* \to \mathcal{N}$ defines a *prefix-code* if its domain is prefix-free. In order to decode a code sequence of a prefix-code, we simply start at the beginning and decode one code word at a time. When we come to the end of a code word, we know it is the end, since no code word is the prefix of any other code word in a prefix-code.

Suppose we encode each binary string $x = x_1 x_2 \ldots x_n$ as

$$\bar{x} = \underbrace{11\ldots1}_{n \text{ times}} 0 x_1 x_2 \ldots x_n.$$

The resulting code is prefix because we can determine where the code word $\bar{x}$ ends by reading it from left to right without backing up. Note $l(\bar{x}) = 2n + 1$; thus, we have encoded strings in $\{0,1\}^*$ in a prefix manner at the price of doubling their length. We can get a much more efficient code by applying the construction above to the length $l(x)$ of x rather than x itself: define $x' = \overline{l(x)}x$, where $l(x)$ is interpreted as a binary string according to the correspondence (1). Then the code D' with $D'(x') = x$ is a prefix code satisfying, for all $x \in \{0,1\}^*$, $l(x') = n + 2\log n + 1$ (here we ignore the 'rounding error' in Equation 2). D' is used throughout this paper as a standard code to encode natural numbers in a prefix free-manner; we call it the *standard prefix-code for the natural numbers*. We use $L_{\mathcal{N}}(x)$ as notation for $l(x')$. When x is interpreted as a number (using the correspondence (1) and (2)), we see that $L_{\mathcal{N}}(x) = \log x + 2 \log \log x + 1$.

Prefix codes and the Kraft inequality Let $\mathcal{X}$ be the set of natural numbers and consider the straightforward non-prefix representation

(1). There are two elements of $\mathcal{X}$ with a description of length 1, four with a description of length 2 and so on. However, for a prefix code D for the natural numbers there are less binary prefix code words of each length: if x is a prefix code word then no $y = xz$ with $z \neq \epsilon$ is a prefix code word. Asymptotically there are less prefix code words of length n than the 2^n source words of length n. Quantification of this intuition for countable $\mathcal{X}$ and arbitrary prefix-codes leads to a precise constraint on the number of code-words of given lengths. This important relation is known as the *Kraft Inequality* and is due to L.G. Kraft.

THEOREM 2.1 (Kraft inequality). *Let $l_1, l_2, \ldots$ be a finite or infinite sequence of natural numbers. There is a prefix-code with this sequence as lengths of its binary code words iff*

$$\sum_n 2^{-l_n} \leq 1.$$

Uniquely Decodable Codes We want to code elements of $\mathcal{X}$ in a way that they can be uniquely reconstructed from the encoding. Such codes are called 'uniquely decodable'. Every prefix-code is a uniquely decodable code. For example, let $\mathcal{X} = \{1,2,3,4\}^*$. If $E(1) = 0$, $E(2) = 10$, $E(3) = 110$, $E(4) = 111$ then 1421 is encoded as 0111100, which can be easily decoded from left to right in a unique way.

On the other hand, not every uniquely decodable code satisfies the prefix condition. Prefix-codes are distinguished from other uniquely decodable codes by the property that the end of a code word is always recognizable as such. This means that decoding can be accomplished without the delay of observing subsequent code words, which is why prefix-codes are also called instantaneous codes.

There is good reason for our emphasis on prefix-codes. Namely, it turns out that Theorem 2.1 stays valid if we replace "prefix-code" by "uniquely decodable code." This important fact means that every uniquely decodable code can be replaced by a prefix-code without changing the set of code-word lengths. In Shannon's and Kolmogorov's theories, we are only interested in code word *lengths* of uniquely decodable codes rather than actual encodings. By the previous argument, we may restrict the set of codes we work with to prefix codes, which are much easier to handle.

Probability distributions and complete prefix codes A uniquely decodable code is *complete* if the addition of any new code word to its code word set results in a non-uniquely decodable code. It is easy to see that a code is complete iff equality holds in the associated Kraft Inequality.

Let $l_1, l_2, \ldots$ be the code words of some complete uniquely decodable code. Let us define $q_x = 2^{-l_x}$. By definition of completeness, we have $\sum_x q_x = 1$. Thus, the q_x can be thought of as *probability mass functions* corresponding to some probability distribution Q. We say Q is the distribution *corresponding* to $l_1, l_2, \ldots$. In this way, each complete uniquely decodable code is mapped to a unique probability distribution. Of course, this is nothing more than a formal correspondence: we may choose to encode outcomes of X using a code corresponding to a distribution Q, whereas the outcomes are actually distributed according to some $P \neq Q$. But, as we show in Theorem 3.5 below, if X is distributed according to P, then the code to which P corresponds is, in an average sense, the code that achieves optimal compression of X.

Prefix codes as protocols for asking questions Prefix codes can be thought of as protocols for sequentially asking yes/no-questions. To make this precise we slightly change our setting. We now think of the 'receiver' B as someone who sequentially asks questions about X. We assume that the 'sender' A only passes on information when asked a question. But in that case, he answers truthfully. The questions of B must all be of the form 'Is the realized value x an element of the set $\mathcal{X}'$?', where $\mathcal{X}'$ is some subset of $\mathcal{X}$. B keeps asking such questions until he has determined the precise value $X = x$. More precisely, B determines a sequence of sets $\mathcal{X}_\epsilon$, $\mathcal{X}_0$, $\mathcal{X}_1$, $\mathcal{X}_{00}$, $\mathcal{X}_{01}$, $\ldots$, satisfying the following two conditions:

1. $\mathcal{X}_\epsilon = \mathcal{X}$.
2. Let $y \in \{0,1\}^*$. If $\mathcal{X}_y$ has more than one element, then $\mathcal{X}_{y0} \cap \mathcal{X}_{y1} = \emptyset$ and $\mathcal{X}_{y0} \cup \mathcal{X}_{y1} = \mathcal{X}_y$. If $\mathcal{X}_y$ has just one element, then $\mathcal{X}_{yz}$ is undefined for any continuation z of y.

The sets $\mathcal{X}_y$ determine B's protocol as follows. First, B asks 'Is $x \in \mathcal{X}_0$?'. If the answer is yes, then B's next question is 'Is $x \in \mathcal{X}_{00}$?' If the answer is no, then B knows that $x \in \mathcal{X}_1$ and B's next question is 'Is $x \in \mathcal{X}_{10}$?' If the answer to the first two questions is yes, B's third question is 'Is $x \in \mathcal{X}_{000}$?' If the answer to the first question is no and to the second yes, then B's question is 'Is $x \in \mathcal{X}_{10}$?', and so on. B keeps asking questions in this way until it has precisely determined the value of x, i.e. until it knows that $x \in \mathcal{X}_y$ for some y such that $\mathcal{X}_y$ has but one element.

To relate such a sequential protocol to prefix codes, consider the code E defined as follows: for all $x \in \mathcal{X}$, we set $E(x) := y$ for the y such that $\mathcal{X}_y = \{x\}$. In this way all $x \in \mathcal{X}$ are assigned a unique code word $E(x)$ such that the set of code words is prefix-free. Therefore, E defines a prefix-code that, for each source word, reserves exactly one code word. Conversely, one can show that each prefix-code that reserves only one code word for each source word coincides with a sequential question-protocol.

Thus, the problems of prefix-free encoding the value of X and sequentially determining (by asking) the value of X are really equivalent. This is yet another reason why prefix codes are more 'natural' than general uniquely decodable codes.

3. Shannon Entropy versus Kolmogorov Complexity

3.1. Shannon Entropy

It seldom happens that a detailed mathematical theory springs forth in essentially final form from a single publication. Such was the case with Shannon information theory, which properly started only with the appearance of C.E. Shannon's paper "The mathematical theory of communication" (Shannon, 1948). In this paper, Shannon proposed a measure of information in a distribution, which he called the 'entropy'. The entropy $H(P)$ of a distribution P measures the 'the inherent uncertainty in P', or (in fact equivalently), 'how much information is gained when an outcome of P is observed'. To make this a bit more precise, let us imagine an observer who knows that X is distributed according to P. The observer then observes $X = x$. The entropy of P stands for the 'uncertainty of the observer about the outcome x *before* he observes it'. Now think of the observer as a 'receiver' who receives the message conveying the value of X. From this dual point of view, the entropy stands for

> the average amount of information that the observer has gained *after* receiving a realized outcome x of the random variable X. $(*)$

Below, we first give Shannon's mathematical definition of entropy, and we then connect it to its intuitive meaning $(*)$.

DEFINITION 3.1. Let $\mathcal{X}$ be a finite or countable set, let X be a random variable taking values in $\mathcal{X}$ with distribution P. Then the (Shannon-) *entropy* of random variable X is given by

$$H(X) = -\sum_{x \in \mathcal{X}} p_x \log p_x, \tag{3}$$

Entropy is defined here as a functional mapping random variables to real numbers. In many texts, entropy is, essentially equivalently, defined as a map from *distributions* of random variables to the real numbers. Thus, by definition: $H(P) := H(X) = -\sum_{x \in \mathcal{X}} p_x \log p_x$.

Motivation Shannon's definition can be motivated in several different ways. The two most important ones are the *axiomatic* approach and the *coding interpretation*. In this paper we concentrate on the latter, but we first briefly sketch the former. The idea of the axiomatic approach is to postulate a small set of eminently reasonable conditions that any measure of information relative to a distribution should satisfy. One then shows that the only measure satisfying all the postulates is the Shannon entropy. We outline this approach for finite sources $\mathcal{X} = \{1, \ldots, N\}$. We look for a function H that maps probability distributions on $\mathcal{X}$ to real numbers. For given distribution P, $H(P)$ should measure 'how much information is gained on average when an outcome is made available'. We can write $H(P) = H(p_1, \ldots, p_N)$ where p_i stands for the probability of i. Suppose we require that

1. $H(p_1, \ldots, p_N)$ is continuous in $p_1, \ldots, p_N$.

2. If all the p_i are equal, $p_i = 1/N$, then H should be a monotonic increasing function of N. With equally likely events there is more choice, or uncertainty, when there are more possible events.

3. If a choice is broken down into two successive choices, the original H should be the weighted sum of the individual values of H. Rather than formalizing this condition, we will give a specific example. Suppose that $\mathcal{X} = \{1, 2, 3\}$, and $p_1 = 1/2, p_2 = 1/3, p_3 = 1/6$. We can think of $x \in \mathcal{X}$ as being generated in a two-stage process. First, an outcome in $\mathcal{X}' = \{0, 1\}$ is generated according to a distribution P' with $p'_0 = p'_1 = 1/2$. If $x' = 1$, we set $x = 1$ and the process stops. If $x' = 0$, then outcome '2' is generated with probability $2/3$ and outcome '3' with probability $1/3$, and the process stops. The final results have the same probabilities as before. In this particular case we require that

$$H(\frac{1}{2}, \frac{1}{3}, \frac{1}{6}) = H(\frac{1}{2}, \frac{1}{2}) + \frac{1}{2} H(\frac{2}{3}, \frac{1}{3}) + \frac{1}{2} H(1).$$

   Thus, the entropy of P must be equal to entropy of the first step in the generation process, plus the weighted sum (weighted according to the probabilities in the first step) of the entropies of the second step in the generation process.

   As a special case, if $\mathcal{X}$ is the n-fold product space of another space $\mathcal{Y}$, $X = (Y_1, \ldots, Y_n)$ and the Y_i are all independently distributed according to P_Y, then $H(P_X) = nH(P_Y)$. For example, the total entropy of n independent tosses of a coin with bias p is $nH(p, 1-p)$.

Remarkably, Shannon (1948) proved that

THEOREM 3.2. *The only H satisfying the three above assumptions is of the form $H = -K \sum_{i=1}^{N} p_i \log p_i$, with K a constant.*

Thus, requirements (1)–(3) lead us to the definition of entropy (3) given above up to an (unimportant) scaling factor. We shall give a concrete interpretation of this factor later on. Besides the defining characteristics (1)–(3), the function H has a few other properties that make it attractive as a measure of information. We mention:

4. $H(p_1, \ldots, p_N)$ is a concave function of the p_i.

5. For each N, H achieves its unique maximum for the uniform distribution $p_i = 1/N$.

6. $H(p_1, \ldots, p_N)$ is zero iff one of the p_i has value 1. Thus, H is zero if and only if we do not gain any information at all if we are told that the outcome is i (since we already knew i would take place with certainty)

We note that there do exist variations of 'entropy' which violate one or more of requirements (1)-(3); a good example is the family of *Rényi entropies* (Cover and Thomas, 1991). While such alternative notions of entropy are useful in their own, restricted context, Shannon's original definition remains by far the most important.

Coding interpretation Immediately after stating Theorem 3.2, Shannon (1948) continues, "this theorem, and the assumptions required for its proof, are in no way necessary for the present theory. It is given chiefly to provide a certain plausibility to some of our later definitions. The *real justification* of these definitions, however, will reside in their implications".

Thus, in the spirit of Shannon, we will henceforth concentrate on a very concrete interpretation of entropy in terms of the length (number of bits) needed to encode outcomes in $\mathcal{X}$. This provides much clearer intuitions; it lies at the root of the many practical applications of information theory, and, most importantly for us, it simplifies the comparison to Kolmogorov complexity.

EXAMPLE 3.3. We start with an example. The entropy of a random variable X with equally likely outcomes in a finite sample space $\mathcal{X}$ is given by $H(X) = \log |\mathcal{X}|$. By choosing a particular message x from $\mathcal{X}$, we remove the entropy from X by the assignment $X := x$ and produce or transmit *information* $I = \log |\mathcal{X}|$ by our selection of x. We show below that $I = \log |\mathcal{X}|$ (or, to be more precise, the integer $I' = \lceil \log |\mathcal{X}| \rceil$) can be interpreted as the number of bits needed to be transmitted from an (imagined) sender to an (imagined) receiver. $\diamondsuit$

We now connect entropy to minimum average code lengths. These are defined as follows:

DEFINITION 3.4. Let source words $x \in \{0,1\}^*$ be produced by a random variable X with probability $P(x) = p_x$ for the event $X = x$. The characteristics of X are fixed. Now consider prefix codes $D : \{0,1\}^* \to \mathcal{N}$ with one code word per source word. and denote the length of the code word for x be l_x. We want to minimize the expected number of bits we have to transmit for the given source X and choose a prefix code D that achieves this. In order to do so, we must minimize the *average code-word length* $\bar{L}_D = \sum_x p_x l_x$. We define the *minimal average code word length* as $\bar{L} = \min\{\bar{L}_D : D \text{ is a prefix-code}\}$. A prefix-code D such that $\bar{L}_D = \bar{L}$ is called an *optimal prefix-code* with respect to prior probability P of the source words.

The (minimal) average code length of an (optimal) code does not depend on the details of the set of code words, but only on the set of code-word lengths. It is just the expected code-word length with respect to the given distribution. Shannon discovered that the minimal average code word length is about equal to the entropy of the source word set. This is known as the *Noiseless Coding Theorem*. The adjective "noiseless" emphasizes that we ignore the possibility of errors.

THEOREM 3.5. *Let $\bar{L}$ and P be as above. If $H(P) = -\sum_x p_x \log p_x$ is the entropy, then*

$$H(P) \leq \bar{L} \leq H(P) + 1. \tag{4}$$

We are typically interested in encoding a binary string of length n with entropy proportional to n (Example 4.1). The essence of (4) is that, for all but the smallest n, the difference between entropy and minimal expected code length is completely negligible.

It turns out that the optimum $\bar{L}$ in (4) is relatively easy to achieve, with the Shannon-Fano code. Let there be N symbols (also called basic messages or source words). Order these symbols according to decreasing probability, say $\mathcal{X} = \{1, 2, \ldots, N\}$ with probabilities $p_1, p_2, \ldots, p_N$. Let $P_r = \sum_{i=1}^{r-1} p_i$, for $r = 1, \ldots, N$. The binary code $E : \mathcal{X} \to \{0,1\}^*$ is obtained by coding r as a binary number $E(r)$, obtained by truncating the binary expansion of P_r at length $l(E(r))$ such that

$$-\log p_r \leq l(E(r)) < 1 - \log p_r.$$

This code is the *Shannon-Fano code*. It has the property that highly probable symbols are mapped to short code words and symbols with low probability are mapped to longer code words (just like in a less optimal setting is done in the Morse code). Moreover,

$$2^{-l(E(r))} \leq p_r < 2^{-l(E(r))+1}.$$

Note that the code for symbol r differs from all codes of symbols $r+1$ through N in one or more bit positions, since for all i with $r+1 \leq i \leq N$,

$$P_i \geq P_r + 2^{-l(E(r))}.$$

Therefore the binary expansions of P_r and P_i differ in the first $l(E(r))$ positions. This means that E is one-to-one, and it has an inverse: the decoding mapping E^{-1}. Even better, since no value of E is a prefix of any other value of E, the set of code words is a prefix-code. This means we can recover the source message from the code message by scanning it from left to right without look-ahead. If H_1 is the average number of bits used per symbol of an original message, then $H_1 = \sum_r p_r l(E(r))$. Combining this with the previous inequality we obtain (4):

$$-\sum_r p_r \log p_r \leq H_1 < \sum_r (1 - \log p_r) p_r = 1 - \sum_r p_r \log p_r.$$

Interpretation in terms of sequential questions We re-interpret Shannon's noiseless coding theorem in terms of protocols for sequentially asking questions: suppose that B asks questions of the type 'Is x in the set $\mathcal{X}'$?', where $\mathcal{X}'$ is some subset of $\mathcal{X}$. A answers truthfully to each question, and B keeps asking questions until he has determined the exact value of the realized outcome x of the random variable X. In Section 2 we showed that each protocol that B can use may be thought of as a prefix code with one code word per source word, and vice versa. Therefore, Theorem 3.5 may be interpreted as follows. Suppose it is B's goal to determine the exact value of X *using as few questions as possible.* If B asks his questions in the cleverest possible way, he will on average need to ask $H(X)$ questions (plus or minus one) to find out the exact value of X. From this point of view, the Shannon-Fano code we described above is *a protocol for asking questions that is 'almost' optimal*, where the 'optimal' protocol is the protocol that minimizes the expected number of questions to be asked.

Problem and Lacuna Shannon observes, "Messages have *meaning* [... however ...] the semantic aspects of communication are irrelevant to the engineering problem." Thus, in Shannon's theory 'information' is fully determined by the probability distribution on the set of possible messages, and unrelated to the meaning, structure or content of individual messages. This is problematic in at least two ways:

First, in many practical cases, the distribution generating outcomes may be unknown to the observer or (worse), may not exist at all[1]. For example, can we answer a question like "what is the information in this book" by viewing it as an element of a set of possible books with a probability distribution on it? This seems unlikely. And how to measure the quantity of hereditary information in biological organisms, as encoded in DNA? Again there is the possibility of seeing a particular form of animal as one of a set of possible forms with a probability distribution on it. This seems to be contradicted by the fact that the calculation of all possible lifeforms in existence at any one time on earth would give a ridiculously low figure like 2^{100}.

Shannon's classical information theory assigns a quantity of information to an ensemble of possible messages. All messages in the ensemble being equally probable, this quantity is the number of bits needed to count all possibilities. This expresses the fact that each message in the ensemble can be communicated using this number of bits. However, it does not say anything about the number of bits needed to convey any individual message in the ensemble, and this constitutes a second 'lacuna' of Shannon's theory. To illustrate this, consider the ensemble consisting of all binary strings of length 9999999999999999.

By Shannon's measure, we require 9999999999999999 bits on the average to encode a string in such an ensemble. However, the string consisting of 9999999999999999 1's can be encoded in about 55 bits by expressing 9999999999999999 in binary and adding the repeated pattern "1." A requirement for this to work is that we have agreed on an algorithm that decodes the encoded string. We can compress the string still further when we note that 9999999999999999 equals $3^2 \times 1111111111111111$, and that 1111111111111111 consists of 2^4 1's.

Thus, we have discovered an interesting phenomenon: the description of some strings can be compressed considerably, provided they exhibit enough regularity. However, if regularity is lacking, it becomes more cumbersome to express large numbers. For instance, it seems easier to compress the number "one billion," than the number "one billion seven hundred thirty-five million two hundred sixty-eight thousand and three hundred ninety-four," even though they are of the same order of magnitude.

We are interested in a measure of information that, unlike Shannon's, does not rely on (often untenable) probabilistic assumptions, and that takes into account the phenomenon that 'regular' strings are compressible. Thus, we aim for a measure of information content of

[1] Even if we adopt a Bayesian (subjective) interpretation of probability, this problem remains (Grünwald, 2003).

an *individual finite object*, and in the information conveyed about an individual finite object by another individual finite object. Here, we want the information content of an object x to be an attribute of x *alone*, and not to depend on, for instance, the means chosen to describe this information content. Surprisingly, this turns out to be possible, at least to a large extent. The resulting theory of information is based on Kolmogorov complexity, a notion independently proposed by Solomonoff (1964), Kolmogorov (1965) and Chaitin (1969); Li and Vitányi (1997) describe the history of the subject.

3.2. Kolmogorov Complexity

Suppose we want to describe a given object by a finite binary string. We do not care whether the object has many descriptions; however, each description should describe but one object. From among all descriptions of an object we can take the length of the shortest description as a measure of the object's complexity. It is natural to call an object "simple" if it has at least one short description, and to call it "complex" if all of its descriptions are long.

As in Section 2, consider a description method D, to be used to transmit messages from a sender to a receiver. If D is known to both a sender and receiver, then a message x can be transmitted from sender to receiver by transmitting the description y with $D(y) = x$. The cost of this transmission is measured by $l(y)$, the length of y. The least cost of transmission of x is determined by the length function $L(x)$: recall that $L(x)$ is the length of the shortest y such that $D(y) = x$. We choose this length function as the descriptional complexity of x under specification method D.

Obviously, this descriptional complexity of x depends crucially on D. The general principle involved is that the syntactic framework of the description language determines the succinctness of description.

In order to objectively compare descriptional complexities of objects, to be able to say "x is more complex than z," the descriptional complexity of x should depend on x alone. This complexity can be viewed as related to a universal description method that is a priori assumed by all senders and receivers. This complexity is optimal if no other description method assigns a lower complexity to any object.

We are not really interested in optimality with respect to all description methods. For specifications to be useful at all it is necessary that the mapping from y to $D(y)$ can be executed in an effective manner. That is, it can at least in principle be performed by humans or machines. This notion has been formalized as that of "partial recursive functions", also known simply as *computable* functions. According to generally accepted mathematical viewpoints it coincides with the intuitive notion of effective computation.

The set of partial recursive functions contains an optimal function that minimizes description length of every other such function. We denote this function by D_0. Namely, for any other recursive function D, for all objects x, there is a description y of x under D_0 that is shorter than any description z of x under D. (That is, shorter up to an additive constant that is independent of x.) Complexity with respect to D_0 minorizes the complexities with respect to all partial recursive functions.

We identify the length of the description of x with respect to a fixed specification function D_0 with the "algorithmic (descriptional) complexity" of x. The optimality of D_0 in the sense above means that the complexity of an object x is invariant (up to an additive constant independent of x) under transition from one optimal specification function to another. Its complexity is an objective attribute of the described object alone: it is an intrinsic property of that object, and it does not depend on the description formalism. This complexity can be viewed as "absolute information content": the amount of information that needs to be transmitted between all senders and receivers when they communicate the message in absence of any other a priori knowledge that restricts the domain of the message. Thus, we have outlined the program for a general theory of algorithmic complexity. The three major innovations are as follows:

1. In restricting ourselves to formally effective descriptions, our definition covers every form of description that is intuitively acceptable as being effective according to general viewpoints in mathematics and logic.

2. The restriction to effective descriptions entails that there is a universal description method that minorizes the description length or complexity with respect to any other effective description method. Significantly, this implies Item 3.

3. The description length or complexity of an object is an intrinsic attribute of the object independent of the particular description method or formalizations thereof.

3.3. Formal Details

The Kolmogorov complexity $K(x)$ of a finite object x will be defined as the length of the shortest effective binary description of x. Broadly speaking, $K(x)$ may be thought of as the length of the shortest computer program that prints x and then halts. This computer program may be written in C, Java, LISP or any other universal language: we shall see that, for any two universal languages, the resulting program lengths differ at most by a constant not depending on x.

To make this precise, let $T_1, T_2, \ldots$ be a standard enumeration of all Turing machines, and let $\phi_1, \phi_2, \ldots$ be the enumeration of corresponding functions which are computed by the respective Turing machines. That is, T_i computes ϕ_i. These functions are the *partial recursive* functions or *computable* functions. For technical reasons we are interested in the so-called prefix complexity, which is associated with Turing machines for which the set of programs (inputs) resulting in a halting computation is prefix free[2]. We can realize this by equipping the Turing machine with a one-way input tape, a separate work tape, and a one-way output tape. Such Turing machines are called prefix machines since the halting programs for any one of them form a prefix free set.

We first define $K_{T_i}(x)$, the prefix Kolmogorov complexity of x relative to a given prefix machine T_i, where T_i is the i-th prefix machine in a standard enumeration of them. $K_{T_i}(x)$ is defined as the length of the shortest input sequence y such that $T_i(y) = \phi_i(y) = x$. If no such input sequence exists, $K_{T_i}(x)$ remains undefined. Of course, this preliminary definition is still highly sensitive to the particular prefix machine T_i that we use. But now the 'universal prefix machine' comes to our rescue. Just as there exists universal ordinary Turing machines, there also exist universal prefix machines. These have the remarkable property that they can simulate every other prefix machine. More specifically, there exists a prefix machine U such that, with as input the pair $\langle i, y \rangle$, it outputs $\phi_i(y)$ and then halts. We now fix, once and for all, a prefix machine U with this property and call U the *reference machine*. The Kolmogorov complexity $K(x)$ of x is defined as $K_U(x)$.

Let us formalize this definition. Let $\langle \cdot \rangle$ be a standard invertible effective one-one encoding from $\mathcal{N} \times \mathcal{N}$ to a prefix-free subset of $\mathcal{N}$. $\langle \cdot \rangle$ may be thought of as the encoding function of a prefix code. For example, we can set $\langle x, y \rangle = x'y'$.

We insist on prefix-freeness and recursiveness because we want a universal Turing machine to be able to read an image under $\langle \cdot \rangle$ from left to right and determine where it ends.

DEFINITION 3.6. *Let U be our reference prefix machine, i.e. for all $i \in \mathcal{N}, y \in \{0,1\}^*$, $U(\langle i, y \rangle) = \phi_i(y)$. The* prefix Kolmogorov

[2] There exists a version of Kolmogorov complexity corresponding to programs that are not necessarily prefix-free, but we will not go into it here.

complexity *of* x *is*

$$K(x) = \min_z \{l(z) : U(z) = x, z \in \{0,1\}^*\} = \\ = \min_{i,y} \{l(\langle i, y \rangle) : \phi_i(y) = x, y \in \{0,1\}^*, i \in \mathcal{N}\}. \quad (5)$$

We can alternatively think of z as a program that prints x and then halts, or as $z = \langle i, y \rangle$ where y is a program such that, when T_i is input program y, it prints x and then halts.

Thus, by definition $K(x) = l(x^*)$, where x^* is the lexicographically first shortest self-delimiting (prefix) program for x with respect to the reference prefix machine. Consider the mapping E^* defined by $E^*(x) = x^*$. This may be viewed as the encoding function of a prefix-code (decoding function) D^* with $D^*(x^*) = x$. By its definition, D^* is a very parsimonious code. The reason for working with prefix rather than standard Turing machines is that, for many of the subsequent developments, we need D^* to be prefix.

Though defined in terms of a particular machine model, the Kolmogorov complexity is machine-independent up to an additive constant and acquires an asymptotically universal and absolute character through Church's thesis, from the ability of universal machines to simulate one another and execute any effective process. The Kolmogorov complexity of an object can be viewed as an absolute and objective quantification of the amount of information in it.

EXAMPLE 3.7. To develop some intuitions, it is useful to think of $K(x)$ as the shortest program for x in some standard programming language such as LISP or Java. Consider the lexicographical enumeration of all syntactically correct LISP programs $\lambda_1, \lambda_2, \ldots$, and the lexicographical enumeration of all syntactically correct Java programs $\pi_1, \pi_2, \ldots$. We assume that both these programs are encoded in some standard prefix-free manner. With proper definitions we can view the programs in both enumerations as computing partial recursive functions from their inputs to their outputs. Choosing reference machines in both enumerations we can define complexities $K_{\text{LISP}}(x)$ and $K_{\text{Java}}(x)$ completely analogous to $K(x)$. All of these measures of the descriptional complexities of x coincide up to a fixed additive constant. Let us show this directly for $K_{\text{LISP}}(x)$ and $K_{\text{Java}}(x)$. Since LISP is universal, there exists a LISP program λ_P implementing a Java-to-LISP compiler. λ_P translates each Java program to an equivalent LISP program. Consequently, for all x, $K_{\text{LISP}}(x) \leq K_{\text{Java}}(x) + 2l(P)$. Similarly, there is a Java program π_L that is a LISP-to-Java compiler, so that for all x, $K_{\text{Java}}(x) \leq K_{\text{LISP}}(x) + 2l(L)$. It follows that $|K_{\text{Java}}(x) - K_{\text{LISP}}(x)| \leq 2l(P) + 2l(L)$ for all x!

The programming language view immediately tells us that $K(x)$ must be small for 'simple' or 'regular' objects x. For example, there exists a fixed-size program that, when input n, outputs the first n bits of π and then halts. Specification of n takes at most $L_{\mathcal{N}}(n) = \log n + 2 \log \log n + 1$ bits. Thus, if x consists of the first n binary digits of π, then $K(x) \overset{+}{<} \log n + 2 \log \log n$. Similarly, if 0^n denotes the string consisting of n 0's, then $K(0^n) \overset{+}{<} \log n + 2 \log \log n$.

On the other hand, for all x, there exists a program 'print x; halt'. This shows that for all $K(x) \overset{+}{<} l(x)$. As was previously noted, for any prefix code, there are no more than 2^m strings x which can be described by m or less bits. In particular, this holds for the prefix code E^* whose length function is $K(x)$. Thus, the fraction of strings x of length n with $K(x) \leq m$ is at most 2^{m-n}: the overwhelming majority of sequences cannot be compressed by more than a constant. Specifically, if x is determined by n independent tosses of a fair coin, then with overwhelming probability, $K(x) \approx l(x)$. Thus, while for very regular strings, the Kolmogorov complexity is small (sublinear in the length of the string), *most* strings are 'random' and have Kolmogorov complexity about equal to their own length. $\diamondsuit$

Problem and Lacuna Unfortunately $K(x)$ is not a recursive function: the Kolmogorov complexity is not computable in general. This means that there exists no computer program that, when input an arbitrary string, outputs the Kolmogorov complexity of that string and then halts. While there exist 'feasible', resource-bounded forms of Kolmogorov complexity (Li and Vitányi 1997), these lack some of the elegant properties of the original, uncomputable notion.

Now suppose we are interested in efficient storage and transmission of long sequences of data. According to Kolmogorov, we can compress such sequences in an essentially optimal way by storing or transmitting the shortest program that generates them. Unfortunately, as we have just seen, we cannot find such a program in general. According to Shannon, we can compress such sequences optimally in an average sense (and therefore, it turns out, also with high probability) if they are distributed according to some P and we know P. Unfortunately, in practice, P is often unknown or even nonexistent. Thus, both Shannon's and Kolmogorov's idea are not directly applicable to most actual data compression problems. For these, we can use *universal codes* which may be viewed at the same time as an extension of Shannon's, and a 'downscaling' of Kolmogorov's theory.

4. Universal Coding: interpolating between Kolmogorov and Shannon

Below we repeatedly use the coding concepts introduced in Section 2. Suppose we are given a recursive enumeration of prefix codes $D_1, D_2, \ldots$. Let $L_1, L_2, \ldots$ be the length functions associated with these codes. That is, $L_i(x) = \min\{l(y) : D_i(y) = x\}$; if there exists no y with $D_i(y) = x$, then $L_i(y) = \infty$. We may encode x by first encoding a natural number k using the standard prefix code for the natural numbers. We then encode x itself using the code D_k. This leads to a so-called *two-part code* $\tilde{D}$ with lengths $\tilde{L}$. By construction, this code is prefix and its lengths satisfy

$$\tilde{L}(x) := \min_{k \in \mathcal{N}} \; L_{\mathcal{N}}(k) + L_k(x), \tag{6}$$

Let $\mathbf{x}$ be an infinite binary sequence and let $x_{[1:n]} \in \{0,1\}^n$ be the initial n-bit segment of this sequence. Since $L_{\mathcal{N}}(k) = O(\log k)$, we have for all k, all n:

$$\tilde{L}(x_{[1:n]}) \leq L_k(x_{[1:n]}) + O(\log k).$$

Recall that for each fixed L_k, the fraction of sequences of length n that can be compressed by more than m bits is less than 2^{-m}. Thus, typically, the codes L_k and the strings $x_{[1:n]}$ will be such that $L_k(x_{[1:n]})$ grows *linearly* with n. This implies that for every $\mathbf{x}$, the newly constructed $\tilde{L}$ is 'almost as good' as whatever code D_k in the list is best for that particular $\mathbf{x}$: the difference in code lengths is bounded by a constant depending on k but not on n. In particular, for each k and each infinite sequence $\mathbf{x}$,

$$\lim_{n \to \infty} \frac{\tilde{L}(x_{[1:n]})}{L_k(x_{[1:n]})} \leq 1. \tag{7}$$

A code satisfying (7) is called a *universal code* relative to the *comparison class* of codes $\{D_1, D_2, \ldots\}$. It is 'universal' in the sense that it compresses every sequence essentially as well as the D_k that compresses that particular sequence the most. In general, there exist many types of codes that are universal: the 2-part universal code defined above is just one means of achieving (7).

Universal codes and Kolmogorov In most practically interesting cases we may assume that for all k, the decoding function D_k is computable, i.e. there exists a prefix Turing machine which for all $y \in \{0,1\}^*$, when input y' (the prefix-free version of y), outputs $D_k(y)$ and then halts. Since such a program has finite length, we must have for all k,

$$l(E^*(x_{[1:n]})) = K(x_{[1:n]}) \leq^+ L_k(x_{[1:n]})$$

where E^* is the encoding function defined earlier, with $l(E^*(x)) = K(x)$. Comparing with (7) shows that the code D^* with encoding function E^* is a universal code relative to $D_1, D_2, \ldots$. Thus, we see that the Kolmogorov complexity K is just the length function of the universal code D^*. Note that D^* is an example of a universal code that is not (explicitly) two-part.

EXAMPLE 4.1. Let us create a universal two-part code that allows us to significantly compress all binary strings with frequency of 0's deviating significantly from 1/2. For $n_0 < n_1$, let $D_{\langle n,n_0 \rangle}$ be the code that assigns code words of equal (minimum) length to all strings of length n with n_0 zeroes, and no code words to any other strings. Then $D_{\langle n,n_0 \rangle}$ is a prefix-code and $L_{\langle n,n_0 \rangle}(x) = \lceil \log \binom{n}{n_0} \rceil$. The universal two part code $\tilde{D}$ relative to the set of codes $\{D_{\langle i,j \rangle} \, : \, i,j \in \mathcal{N}\}$ then achieves the following lengths (to within 1 bit): for all n, all $n_0 \in \{0, \ldots, n\}$, all $x_{[1:n]}$ with n_0 zeroes,

$$\begin{aligned} \tilde{L}(x_{[1:n]}) &= \log n + \log n_0 + 2 \log \log n + 2 \log \log n_0 + \log \binom{n}{n_0} \\ &= \log \binom{n}{n_0} + O(\log n). \end{aligned} \tag{8}$$

Using Stirling's approximation of the factorial, $n! \sim n^n e^{-n} \sqrt{2\pi n}$, we find that

$$\begin{aligned} \log \binom{n}{n_0} &= \log n! - \log n_0! + \log(n - n_0)! = \\ &= n \log n - n_0 \log n_0 - (n - n_0) \log(n - n_0) + O(\log n) \\ &= nH(n_0/n) + O(\log n). \end{aligned} \tag{9}$$

Note that $H(n_0/n) \leq 1$, with equality iff $n_0 = n$. Therefore, if the frequency deviates significantly from 1/2, $\tilde{D}$ compresses $x_{[1:n]}$ by an factor linear in n. In all such cases, D^* compresses the data by at least the same linear factor. Note that (a) each individual code $D_{\langle n,n_0 \rangle}$ is capable of exploiting a particular type of regularity in a sequence to compress that sequence, (b) the universal code $\tilde{D}$ may exploit *many* different types of regularities to compress a sequence, and (c) the code D^* with lengths given by the Kolmogorov complexity asymptotically exploits *all* computable regularities so as to maximally compress a sequence. $\diamondsuit$

Universal codes and Shannon If X is distributed according to some distribution P, then the optimal (in the average sense) code to use is the Shannon-Fano code. But now suppose it is only known that $P \in \mathcal{P}$, where $\mathcal{P}$ is some given (possibly very large, e.g. uncountable) set of candidate distributions. Now it is not clear what code is optimal. We may try the Shannon-Fano code for a particular $P \in \mathcal{P}$, but such a code will typically lead to very large expected code lengths if X turns out to be distributed according to some $Q \in \mathcal{P}, Q \neq P$. We may ask whether there exists another code that is 'almost' as good as the Shannon-Fano code for P, no matter what $P \in \mathcal{P}$ actually generates the sequence? We now show that, provided $\mathcal{P}$ is finite or countable, then (perhaps surprisingly), the answer is yes. To see this, we need the notion of an *information source*. An information source may be thought of as a probability distribution over arbitrarily long sequences, of which an observer gets to see longer and longer initial segments; examples are given below. Formally, an information source P is a probability distribution on the set $\{0,1\}^\infty$ of one-way infinite sequences. Such a P can be identified with the distributions $P^{(1)}$ on $\{0,1\}^1$, $P^{(2)}$ on $\{0,1\}^2, \ldots$. Here $P^{(n)}$ denotes the *marginal* distribution of P on the first n-bit segments. $P^{(n)}$ is related to $P^{(n+1)}$ as follows: for all $n \geq 0$, all $x \in \{0,1\}^n$, $\sum_{y \in \{0,1\}} P^{(n+1)}(xy) = P^{(n)}(x)$ and $P^{(0)}(x) = 1$.

Suppose then that $\mathcal{P}$ is a finite or countable set of information sources. Then the members of $\mathcal{P}$ may be listed as $P_1, P_2, \ldots$. To each marginal distribution $P_k^{(n)}$, there corresponds a unique Shannon-Fano code defined on the set $\{0,1\}^n$ with lengths $L_{\langle n,k \rangle}(x) := \lceil -\log P_k^{(n)}(x) \rceil$.

For given $P \in \mathcal{P}$, we define

$$H(P^{(n)}) := \sum_{x \in \{0,1\}^n} P^{(n)}(x)[-\log P^{(n)}(x)]$$

as the entropy of the distribution of the first n outcomes.

Let E be a prefix-code assigning codeword $E(x)$ to source word $x \in \{0,1\}^n$. The Noiseless Coding Theorem 3.5 on page 12 asserts that the minimal average codeword length $\bar{L}(P) = \sum_{x \in \{0,1\}^n} P(x) l(E(x))$ among all such prefix-codes E satisfies

$$H(P^{(n)}) \leq L(P) \leq H(P^{(n)}) + 1.$$

The entropy $H(P^{(n)})$ can therefore be interpreted as the expected code length of encoding the first n bits generated by the source P, when the optimal (Shannon-Fano) code is used.

We look for a prefix code $\tilde{D}$ with length function $\tilde{L}$ that satisfies, for all $P \in \mathcal{P}$:

$$\lim_{n \to \infty} \frac{\mathbf{E}_P \tilde{L}(X_{[1:n]})}{H(P^{(n)})} \leq 1. \tag{10}$$

where $\mathbf{E}_P \tilde{L}(X_{[1:n]}) = \sum_{x \in \{0,1\}^n} P^{(n)}(x) L(x)$. Define $\tilde{D}$ as the following two-part code: first, n is encoded using the standard prefix code for natural numbers. Then, among all codes $D_{\langle n,k\rangle}$, the k that minimizes $L_{\langle n,k\rangle}(x)$ is encoded (again using the standard prefix code); finally, x is encoded in $L_{\langle n,k\rangle}(x)$ bits. Then for all n, for all k, for *every* sequence $x_{[1:n]}$,

$$\tilde{L}(x_{[1:n]}) \leq L_{\langle n,k\rangle}(x_{[1:n]}) + L_{\mathcal{N}}(k) + L_{\mathcal{N}}(n) \tag{11}$$

Since (11) holds for all strings of length n, it must also hold in expectation for all possible distributions on strings of length n. In particular, this gives, for all $k \in \mathcal{N}$,

$$\mathbf{E}_{P_k}\tilde{L}(X_{[1:n]}) \leq \mathbf{E}_{P_k}L_{\langle n,k\rangle}(X_{[1:n]}) + O(\log n) = H(P_k^{(n)}) + O(\log n),$$

from which (10) follows.

Historically, codes satisfying (10) have been called *universal codes* relative to $\mathcal{P}$; codes satisfying (7) have been considered in the literature only much more recently and are usually called 'universal codes for individual sequences' (Merhav and Feder, 1998). The two-part code $\tilde{D}$ that we just defined is universal both in an individual sequence and in an average sense: $\tilde{D}$ achieves code lengths within a constant of that achieved by $D_{\langle n,k\rangle}$ for *every individual sequence*, for *every* $k \in \mathcal{N}$; but $\tilde{D}$ also achieves expected code lengths within a constant of the Shannon-Fano code for P, for *every* $P \in \mathcal{P}$. We may say that $\tilde{D}$ *interpolates* between Shannon's codes, which are optimal for a specific P, and Kolmogorov's code D^* (with length function K), which by definition does at least as well (within an additive constant) as $\tilde{D}$.

EXAMPLE 4.2. Suppose our sequence is generated by independent tosses of a coin with bias p of tossing "head" where $p \in (0,1)$. Identifying 'heads' with 1, the probability of $n - n_0$ outcomes "1" in an initial segment $x_{[1:n]}$ is then $(1-p)^{n_0}p^{n-n_0}$. Let $\mathcal{P}$ be the set of corresponding information sources, containing one element for each $p \in (0,1)$. $\mathcal{P}$ is an uncountable set; nevertheless, a universal code for $\mathcal{P}$ exists. In fact, it can be shown that the code $\tilde{D}$ with lengths (9) in Example 4.1 is universal for $\mathcal{P}$, i.e. it satisfies (10). The reason for this is (roughly) as follows: if data are generated by a coin with bias p, then with probability 1, the frequency n_0/n converges to p, so that, by (9), $n^{-1}\tilde{L}(x_{[1:n]})$ tends to $n^{-1}H(P^{(n)}) = H(p, 1-p)$.

If we are interested in practical data-compression, then the assumption that the data are generated by a biased-coin source is very restricted. But there are much richer classes of distributions $\mathcal{P}$ for which we can formulate universal codes. For example, we can take $\mathcal{P}$ to be the class of all Markov sources of each order; here the probability that $X_i = 1$ may depend on arbitrarily many earlier outcomes. Such ideas form the basis of most data compression schemes used in practice.

Codes which are universal for the class of all Markov sources of each order and which encode and decode in real-time can easily be implemented. Thus, while we cannot find the shortest program that generates a particular sequence, it is often possible to effectively find the shortest encoding within a quite sophisticated class of codes. ◇

Expected Kolmogorov Complexity $\stackrel{\pm}{=}$ Shannon Entropy: Suppose the source words x are distributed as a random variable X with probability $P(x)$. While $K(x)$ is fixed for each x and gives the shortest code word length (but only up to a fixed constant) and is *independent* of the probability distribution P, we may wonder whether K is also universal in the following sense: If we weigh each individual code word length for x with its probability $P(x)$ the resulting P-expected code word length $\sum_x P(x)K(x)$ achieves the minimal average code word length $H(P) = -\sum_x P(x)\log P(x)$? Here we sum over the entire support of P; restricting summation to a small set, for example the singleton set $\{x\}$, can give a different result. The reasoning above implies that, under some mild restrictions on the distributions P, the answer is yes. This is expressed in the following theorem, where, instead of the quotient we look at the difference of $\sum_x P(x)K(x)$ and $H(P)$. This allows us to express really small distinctions. We call an informatin source P *recursive* if there exists a Turing machine that, when input $\langle n, x, y\rangle$ with $x \in \{0,1\}^*$ and $y, n \in \mathcal{N}$, outputs $P^{(n)}(x)$ to precision $1/y$. The following theorem can be found in (Li and Vitányi, 1997):

THEOREM 4.3. *Let P be a recursive information source. Then for all n,*

$$0 \leq \sum_{x \in \{0,1\}^n} P^{(n)}(x)K(x) - H(P^{(n)}) \leq c_P$$

where c_P is a constant that depends only on P (and not on n).

The Shannon-Fano code for a computable distribution is itself computable. Therefore, for every computable distribution P, the universal code D^* whose length function is the Kolmogorov complexity compresses on average at least as much as the Shannon-Fano code for P. This is the intuitive reason why, no matter what computable distribution P we take, its expected Kolmogorov complexity is close to its entropy.

5. Mutual Information

5.1. Shannon Mutual Information

How much information can a random variable X convey about a random variable Y? Taking a purely combinatorial approach, this notion is captured as follows: If X ranges over S_X and Y ranges over S_Y, then we look at the set U of possible events $(X = a, Y = b)$ consisting of joint occurrences of event $X = a$ and event $Y = b$. If U does not equal the Cartesian product $S_X \times S_Y$, then this means there is some dependency between X and Y. Considering the set $U_a = \{(a, y) : (a, y) \in U\}$ for $a \in S_X$, it is natural to define the *conditional entropy* of Y given $X = a$ as $H(Y|X = a) = \log d(U_a)$. This suggests immediately that the information given by $X = a$ about Y is

$$I(X = a : Y) = H(Y) - H(Y|X = a).$$

For example, if $U = \{(1, 1), (1, 2), (2, 3)\}$, $U \subseteq S_X \times S_Y$ with $S_X = \{1, 2\}$ and $S_Y = \{1, 2, 3, 4\}$, then $I(X = 1 : Y) = 1$ and $I(X = 2 : Y) = 2$.

In this formulation it is obvious that $H(X|X = a) = 0$, and that $I(X = a : X) = H(X)$. This approach amounts to the assumption of *uniform distribution* of the probabilities concerned.

We can generalize this approach, taking into account the frequencies or probabilities of the occurrences of the different values X and Y can assume. Let the *joint probability* $p(a, b)$ be defined as: "the probability of the joint occurrence of event $X = a$ and event $Y = b$." This leads to the self-evident formulas for joint variables X, Y:

$$\begin{aligned}
H(X, Y) &= -\sum_{a,b} p(a, b) \log p(a, b), \\
H(X) &= -\sum_{a,b} p(a, b) \log \sum_{b} p(a, b), \\
H(Y) &= -\sum_{a,b} p(a, b) \log \sum_{a} p(a, b),
\end{aligned}$$

where summation over a is taken over all outcomes of the random variable X and summation over b is taken over all outcomes of random variable Y. One can show that

$$H(X, Y) \leq H(X) + H(Y), \tag{12}$$

with equality only in the case that X and Y are independent. In all of these equations the entropy quantity on the left-hand side increases if we choose the probabilities on the right-hand side more equally.

Conditional entropy The conditional probability $p(b|a)$ of outcome $Y = b$ given outcome $X = a$ for random variables X and Y (not necessarily independent) is defined by

$$p(b|a) = \frac{p(a,b)}{\sum_b p(a,b)},$$

This leads to the following analysis of the information in X about Y by first considering the *conditional entropy* of Y given X as the average of the entropy for Y for each value of X weighted by the probability of getting that particular value:

$$\begin{aligned} H(Y|X) &= \sum_a p(a) H(Y|X=a) \\ &= -\sum_a p(a) \sum_b p(b|a) \log p(b|a) \\ &= -\sum_{a,b} p(a,b) \log p(b|a). \end{aligned}$$

The quantity on the left-hand side tells us how uncertain we are about the outcome of Y when we know an outcome of X. With

$$\begin{aligned} H(X) &= -\sum_a p(a) \log p(a) \\ &= -\sum_a \left(\sum_b p(a,b) \right) \log \sum_b p(a,b) \\ &= -\sum_{a,b} p(a,b) \log \sum_b p(a,b), \end{aligned}$$

and substituting the formula for $p(b|a)$, we find $H(Y|X) = H(X,Y) - H(X)$. Rewrite this expression as the Entropy Equality

$$H(X,Y) = H(X) + H(Y|X). \tag{13}$$

This can be interpreted as, "the uncertainty of the joint event (X,Y) is the uncertainty of X plus the uncertainty of Y given X." Combining Equations 12, 13 gives $H(Y) \geq H(Y|X)$, which can be taken to imply that knowledge of X can never increase uncertainty of Y. In fact, uncertainty in Y will be decreased unless X and Y are independent. Finally, the *information* in the outcome $X = a$ about Y is defined as

$$I(X = a : Y) = H(Y) - H(Y|X = a). \tag{14}$$

Here the quantities $H(Y)$ and $H(Y|X = a)$ on the right-hand side of the equations are always equal to or less than the corresponding quantities under the uniform distribution we analyzed first. The values of the quantities $I(X = a : Y)$ under the assumption of uniform distribution of Y and $Y|X = a$ versus any other distribution are not related by inequality in a particular direction. The equalities $H(X|X = a) = 0$ and $I(X = a : X) = H(X)$ hold under any distribution of the variables. Since $I(X = a : Y)$ is a function of outcomes of X, while $I(Y = b : X)$ is a function of outcomes of Y, we do not compare them directly. However, forming the expectation defined as

$$\begin{aligned}
\mathbf{E}(I(X = a : Y)) &= \sum_a p(a) I(X = a : Y), \\
\mathbf{E}(I(Y = b : X)) &= \sum_b p(b) I(Y = b : X),
\end{aligned}$$

and combining Equations 13, 14, we see that the resulting quantities are equal. Denoting this quantity by $I(X;Y)$ and calling it the *mutual information* in X and Y, we see that this information is *symmetric*:

$$I(X;Y) = \mathbf{E}(I(X = a : Y)) = \mathbf{E}(I(Y = b : X)). \tag{15}$$

EXAMPLE 5.1. Suppose we want to exchange the information about the outcome $X = x$ and it is known already that outcome $Y = y$ is the case, that is, x has property y. Then we require (using the Shannon-Fano code) about $-\log P(X = x|Y = y)$ bits to communicate x. On average, over the joint distribution $P(X = x, Y = y)$ we use $H(X|Y)$ bits, which is optimal by Shannon's noiseless coding theorem. In fact, exploiting the mutual information paradigm, the expected information that outcome $Y = y$ gives about outcome $X = x$ is the same as the expected information that $X = x$ gives about $Y = y$. $\diamondsuit$

Interpretation in terms of sequential questions Just as we did for the entropy, we can also re-interpret mutual information in terms of protocols for asking questions. Suppose that B sequentially asks questions about X, but, as in Example 5.1, before he has to ask any questions, B is told that $Y = y$. B then sequentially asks questions to find out the value of X, using the protocol defined by the Shannon-Fano code for $P(X = \cdot \mid Y = y)$. By Shannon's noiseless coding theorem, this is the optimal protocol. Intuitively, since B is given some initial information, we expect that B has to ask fewer questions than if he were *not* given any initial information. $I(Y;X)$ denotes exactly how many fewer questions B can expect to need to ask *on average* if he is already told the value of Y before asking any questions. Here the average is over both X and Y. Indeed, on average, B needs to ask fewer questions, since $I(Y;X) \geq 0$. But there may certainly exist *individual* y such that $I(Y = y : X)$ is negative. For example, we may have $\mathcal{X} = \{0,1\}$, $\mathcal{Y} =$

$\{0,1\}$, $P(X=1|Y=0)=1$, $P(X=1|Y=1)=1/2$, $P(Y=1)=\epsilon$. Then $I(Y;X)=H(\epsilon,1-\epsilon)$ whereas $I(Y=1:X)=H(\epsilon,1-\epsilon)+\epsilon-1$. For small ϵ, this quantity is smaller than 0.

Problem and Lacuna The quantity $I(X;Y)$ symmetrically characterizes to what extent random variables X and Y are correlated. An inherent problem with probabilistic definitions is that — as we have just seen — although $\mathbf{E}(I(Y:X))$ is always positive, for some probability distributions and some y, $I(Y=y:X)$ can turn out to be negative—which definitely contradicts our naive notion of information content. How is this possible? The concept of information as used in the theory of communication is a probabilistic notion, which is natural for information transmission over communication channels. Nonetheless, we tend to identify *probabilities* of messages with *frequencies* of messages in a sufficiently long sequence, which under some conditions on the stochastic source can be rigorously justified. The great probabilist, Kolmogorov, remarks, "If something goes wrong here, the problem lies in the vagueness of our ideas of the relation between mathematical probability theory and real random events in general." The *algorithmic* mutual information we introduce below can *never* be negative, and in this sense is closer to the intuitive notion of information content.

5.2. Algorithmic Mutual Information

Conditional Kolmogorov Complexity To prepare for the definition of Shannon mutual information, we first needed to introduce a conditional version of entropy. Analogously, to prepare for the definition of algorithmic mutual information, we need a notion of conditional Kolmogorov complexity. Intuitively, the conditional prefix Kolmogorov complexity $K(x|y)$ of x given y can be interpreted as the shortest prefix program p such that, when y is given to the program p as input, the program prints x and then halts. The idea of providing p with an input y is realized by putting $\langle p,y\rangle$ rather than just p on the input tape of the universal prefix machine U.

DEFINITION 5.2. *The* conditional prefix Kolmogorov complexity *of x given y (for free) is*

$$K(x|y)=\min_p\{l(p):U(\langle p,y\rangle)=x, p\in\{0,1\}^*\}.$$

We define $K(x)=K(x|\epsilon)$.

Note that we just redefined $K(x)$ so that the unconditional Kolmogorov complexity is *exactly* equal to the conditional Kolmogorov complexity with empty input. This does not contradict our earlier definition: we can choose a reference prefix machine U such that $U(\langle p, \epsilon \rangle) = U(p)$. Then we automatically have $K(x) = K(x|\epsilon)$.

Recall from Section 2 the notation $\stackrel{+}{=}, \stackrel{+}{>}$. By definition, $K(x, y) = K(\langle x, y \rangle)$. Trivially, the symmetry property holds: $K(x, y) \stackrel{+}{=} K(y, x)$. An interesting property is the "Additivity of Complexity" property

$$K(x, y) \stackrel{+}{=} K(x) + K(y \mid x^*) \stackrel{+}{=} K(y) + K(x \mid y^*). \tag{16}$$

where x^* is the first (in standard enumeration order) shortest prefix program that generates x and then halts. It is easy to see that x^* has the same information as the pair $x, K(x)$: given x^* we can compute x and $l(x^*) = K(x)$; given $x, K(x)$ we can run all programs simultaneously in dovetailed fashion and select the first program of length $K(x)$ that halts with output x as x^*. (Dovetailed fashion means that in phase k of the process we run all programs i for j steps such that $i + j = k$, $k = 1, 2, \ldots$) (16) is the Kolmogorov complexity equivalent of the entropy equality (13). That this latter equality holds is true by simply rewriting both sides of the equation according to the definitions of averages of joint and marginal probabilities. In fact, potential individual differences are averaged out. But in the Kolmogorov complexity case we do nothing like that: it is truly remarkable that additivity of algorithmic information holds for individual objects.

The result (16) is due to Gács (1994), can be found as Theorem 3.9.1 in (Li and Vitányi, 1997) and has a difficult proof. It is perhaps instructive to point out that the version with just x and y in the conditionals doesn't hold with $\stackrel{+}{=}$, but holds up to additive logarithmic terms that cannot be eliminated.

To define the algorithmic mutual information between two individual objects x and y with no probabilities involved, it is instructive to first recall the probabilistic notion (15) Rewriting (15) as

$$\sum_x \sum_y p(x, y)[-\log p(x) - \log p(y) + \log p(x, y)],$$

and noting that $-\log p(s)$ is very close to the length of the prefix-free Shannon-Fano code for s, we are led to the following definition. The *information in y about x* is defined as

$$I(y : x) = K(x) - K(x \mid y^*) \stackrel{+}{=} K(x) + K(y) - K(x, y), \tag{17}$$

where the second equality is a consequence of (16) and states that this information is symmetrical, $I(x:y) \stackrel{+}{=} I(y:x)$, and therefore we can talk about *mutual information*.[3]

Theorem 4.3 gave the relationship between entropy and ordinary Kolmogorov complexity; it showed that the entropy of distribution P is approximately equal to the expected (under P) Kolmogorov complexity. Theorem 5.3 gives the analogous result for the mutual information (to facilitate comparison to Theorem 4.3 , note that x and y in (18) below may stand for strings of arbitrary length n).

THEOREM 5.3. *Given a recursive probability mass distribution $p(x,y)$ over (x,y) we have*

$$I(X;Y) - K(p) \stackrel{+}{<} \sum_x \sum_y p(x,y) I(x:y) \stackrel{+}{<} I(X;Y) + 2K(p), \qquad (18)$$

where c_p is a constant that depends only on p (it is the length of the shortest prefix-free program that computes $p(x,y)$ from input (x,y)).

Thus, we see that the expectation of the algorithmic mutual information $I(x:y)$ is close to the probabilistic mutual information $I(X;Y)$.

Interpretation in terms of sequential questions The algorithmic mutual information $I(y:x) = K(x) - K(x|y^*)$ which equals $K(x) - K(x|y)$ up to an additive logarithmic term $O(\log K(y))$ is the savings in number of questions B needs to ask to get to know x if B already knows y. Clearly, if y is the empty word, no information at all, then B needs to ask $K(x)$ yes-no questions to obtain the consecutive bits of x^*. But if B already knows y then he needs to ask only $K(x|y)$ such questions to obtain the shortest program to compute from y to x. The caveat being, as usual, that B has arbitrary amounts of time and storage to perform its computation from x to y. For specific individual x, y this number can be far less than the average as given by Shannon's mutual information.

Problem and Lacuna Entropy, Kolmogorov complexity and mutual (algorithmic) information are concepts that do not distinguish between different *kinds* of information (such as 'meaningful' and 'meaningless'

[3] The notation of the algorithmic (individual) notion $I(x:y)$ distinguishes it from the probabilistic (average) notion $I(X;Y)$. We deviate slightly from (Li and Vitányi, 1997) where $I(y:x)$ is defined as $K(x) - K(x \mid y)$.

information). Such more refined notions can be arrived at by *constraining* the description methods with which strings are allowed to be encoded, and by considering *lossy* rather than *lossless* encoding. Yet the basic notions entropy, Kolmogorov complexity and mutual information continue to play a fundamental rôle. The two most important developments are *rate-distortion theory* in the Shannon setting (Shannon, 1948, Cover and Thomas, 1991), dealing with 'useful' information, and the *Kolmogorov structure function* in Kolmogorov's setting, dealing with 'meaningful' information (Kolmogorov, 1974; Shen', 1983; Cover and Thomas, 1991; Gács et al. 2001; Vereshchagin and Vitányi, 2002; Vitányi, 2002; Rissanen, 2002). It is here that the two theories may have something relevant to say about the notions of 'information' that are studied within the logic and semantics of natural language communities (Van Rooy 2003). We briefly illustrate this for the rate-distortion theory.

6. Shannon's Rate Distortion; Information in Questions

As before, we consider a situation in which sender A wants to communicate the outcome of random variable X to receiver B. The distribution of X is known to both A and B. But now A is only allowed to use a finite number, say R bits, to communicate, so that A can only send 2^R different messages. Then the encoding function E has to map $\mathcal{X}$ to $\{0,1\}^R$, and D has to map $\{0,1\}^R$ back to $\mathcal{X}$. If $|\mathcal{X}| > 2^R$ or if $\mathcal{X}$ is uncountable (say, $\mathcal{X} = \mathbb{R}$), then there can be no code (D, E) such that for all x, $D(E(x)) = x$. Thus, A and B cannot make sure that x can always be reconstructed. As the next best thing, they may agree on a code such that for all x, $D(E(x))$ is in some sense 'as close as possible' to the original x. To formalize this for a given code (D, E), we define $\hat{X} : \mathcal{X} \to \mathcal{X}$ as the function $\hat{X}(x) := D(E(x))$, and we let $\hat{\mathcal{X}}$ be the range of $\hat{X}$. We may interpret $\hat{X}(x)$ as an estimate of x, and $\hat{\mathcal{X}}$ as the set of values it can take. We assume that the 'goodness' of $\hat{X}(x)$ as an approximation of x is measured using some *distortion function* $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. This distortion function may be anything that is appropriate to the situation at hand. Once d is fixed, we may consider the *expected* distortion

$$\mathbf{E}[d(X, \hat{X})] = \sum_{x \in \mathcal{X}} p_x d(x, \hat{X}(x)), \tag{19}$$

where, if $\mathcal{X} = \mathbb{R}$, the sum is replaced by an integral and p_x stands for the probability density of x with respect to Lebesgue measure.

In the rate distortion setting, the goal of A and B is to determine the code (D, E) with associated $\hat{X}$ that minimizes the expected distortion.

EXAMPLE 6.1. Suppose X is a real-valued, normally (Gaussian) distributed random variable with mean $\mathbf{E}[X] = 0$ and variance $\mathbf{E}[X - \mathbf{E}[X]]^2 = \sigma^2$. Let us use the squared Euclidean distance $d(x, \hat{x}) = (x - \hat{x})^2$ as a distortion measure. If A is allowed to use R bits, then $\hat{\mathcal{X}}$ can have no more than 2^R elements, whereas $\mathcal{X} = \mathbb{R}$ is uncountably infinite. We should choose $\hat{\mathcal{X}}$ and the function $\hat{X}$ such that (19) is minimized. Suppose first $R = 1$. Then the optimal $\hat{X}$ turns out to be

$$\hat{X}(x) = \begin{cases} \sqrt{\frac{2}{\pi}}\sigma^2 & \text{if } x \geq 0 \\ -\sqrt{\frac{2}{\pi}}\sigma^2 & \text{if } x < 0. \end{cases}$$

Thus, the domain $\mathcal{X}$ is partitioned into two regions, one corresponding to $x \geq 0$, and one to $x < 0$. That the boundary should be at $x = 0$ is evident by the symmetry of the Gaussian distribution around 0. Within each region, one then picks a 'representative point' so as to minimize (19). Similarly, if $R = 2$, then $\mathcal{X}$ should be partitioned into 4 regions, each of which are to be represented by a single point such that (19) is minimized. An extreme case is $R = 0$: how should B estimate X if it is not given any information whatsoever? This means that $\hat{X}(x)$ must take the same value for all x. The expected distortion (19) is then minimized if B picks $\hat{X} \equiv 0$, giving distortion equal to σ^2. $\diamondsuit$

There is no reason in general that the distortion function should be symmetric: in fact, it may be anything that pertains to the situation at hand. It can be considered as (minus) a *utility* function, indicating the loss that B incurs if he has to predict x without knowing its precise value.

Interpretation in terms of sequential questions Previously, we interpreted entropy as the expected minimum number of yes/no-questions that receiver has to ask to sender in order to determine the precise outcome x of a random variable X.

The present setting can be interpreted in terms of a more involved question-and-answer game: now receiver is allowed to ask only R yes/no-questions. He then has to come up with a guess $\hat{x}$ of the outcome x. The quality of this guess is measured by $d(x, \hat{x})$. The goal of the receiver is now to ask the R 'cleverest possible questions' that reduce his expected distortion as much as possible; equivalently, they increase his expected utility as much as possible. Thus, there is a relation to 'quality and quantity of information exchange' (Van Rooy 2003) as studied in natural language semantics.

As a concrete case, if $R = 1$, then in the Gaussian example above, receiver should ask "Is $x \in [0, \infty)$ or not?". Every other question reduces the expected distortion by a lesser amount. In general, the present question-and-answer game is very different from the original game where the goal was to minimize the total number of questions. But the following example shows that, if we take a special distortion measure, then the goal of minimizing distortion and minimizing total number of questions are reconciled.

EXAMPLE 6.2. Suppose receiver wants to estimate the actual x by a probability distribution P on $\mathcal{X}$. Thus, if R bits are allowed to be used, one of R different distributions on $\mathcal{X}$ can be sent to receiver. The best that can be done is to partition $\mathcal{X}$ into 2^R subsets $\mathcal{A}_1, \ldots, \mathcal{A}_{2^R}$. Sender observes the i such that $x \in \mathcal{A}_i$ and passes this information on to receiver. A little thought reveals that the information i tells the receiver that X is now distributed according to the conditional distribution $P(X = \cdot \mid X \in \mathcal{A}_i)$. It is then natural to measure the quality of distribution $P(X = \cdot \mid X \in \mathcal{A}_i)$ by its entropy, i.e. by the additional number of questions that receiver has to ask before he knows the value of x with certainty. That is, we take $d(x, P) = -\log P(x)$: the distortion function is the Shannon-Fano code length for the communicated distribution. Here we implicitly generalized the definition of 'distortion' measure: we do not require the estimates $\hat{X}$ to take values in $\mathcal{X}$ any more. Rather, they are now a set of probability distributions on $\mathcal{X}$; the new definition includes the former as a special case.

With $d(x, P) = -\log P(x)$, the expected distortion is $\mathbf{E}[d(X, P)] = H(P)$. The minimum achievable distortion $d^*(r)$ for $R = r$ is given by

$$d^*(r) = \min I(Y; X)$$

where $\mathcal{Y} = \{1, \ldots, 2^R\}$, and the minimum is over all sets $\mathcal{Y}$ and all distributions P^* over $\mathcal{X} \times \mathcal{Y}$ such that for all $y \in \mathcal{Y}$, $P^*(Y = y) = \sum_x P^*(Y = y, X = x) = P(Y = y)$. In particular, for $r = 0$, $d^*(r) = H(P)$; for $r \geq H(P)$, $d^*(r) = 0$; for general r, $d^*(r)$ is the minimum expected number of questions that B still has to ask to determine x, just after B has already been given the answers to the first r questions.

Thus, if we pick the Shannon-Fano code length as the distortion measure, then the rate-distortion theory is reconciled with the lossless compression theory. In this case, the distortion-rate function $d^*(r)$ shows how fast the entropy decreases (the information gained by receiver increases) if receiver always asks the 'cleverest possible question', that has the highest expected information gain. $\diamondsuit$

Rate distortion and mutual information As R increases, the minimum achievable distortion becomes smaller and smaller. Shannon was interested in studying the functional relationship between R and the minimum achievable distortion d^* for a given R. This is called the *distortion-rate* function. For technical reasons it is often more convenient to study R as a function of d^*. This is the celebrated *rate-distortion* function. As one of the main results in his original paper, Shannon (1948) showed that there is a deep connection between the mutual information and the rate-distortion function which holds *no matter what distortion function d is used* - thus not only for the Shannon-Fano distortion. We only mention this result because it illustrates that mutual information is a fundamental notion; for a precise statement we refer to (Cover and Thomas, 1991).

7. Further Topics and Further Reading

Further Topics: Shannon Of the three most important developments in Shannon's original paper, we only discussed two: the *noiseless coding theorem* for *lossless* compression (Theorem 3.5) and the notion of *rate-distortion* related to *lossy* compression. We did not discuss the *channel coding theorem*, which is related to *lossless* communication over a *noisy* channel. These and many other topics in Shannon information theory are thoroughly discussed and explained in the standard reference (Cover and Thomas, 1991).

Further Topics: Kolmogorov Kolmogorov complexity has many applications which we could not discuss here. It leads to a formal notion of *randomness of individual sequences* that does not refer to an underlying probability distribution. Also, it lies at the basis of a powerful mathematical theory of *inductive inference*. Third, it has led to a new mathematical proof technique called the *incompressibility method*. These and many other topics in Kolmogorov complexity are thoroughly discussed and explained in the standard reference (Li and Vitányi, 1991).

We end by mentioning an exciting recent development: the *Kolmogorov structure function*.

Kolmogorov Structure Function The Kolmogorov Structure Function (Kolmogorov, 1974; Shen', 1983; Cover and Thomas, 1991; Gács et al. 2001; Vereshchagin and Vitányi, 2002; Vitányi, 2002; Rissanen, 2002) can be viewed (to some extent) as the analogue in Kolmogorov's theory of Shannon's rate distortion. It is based on encoding objects (strings) in two parts: a *structural* and a *random* part. We encountered a very simple example of such a description in Example 4.1, where we first encoded the frequency of ones in a string (a very simple 'structure') and then the particular sequence with the given frequency (corresponding to the 'random' part of the description). Intuitively, the 'meaning' of the string resides in the structural part and the size of the structural part quantifies the 'meaningful' information in the message. Recently, there have been many exciting new results in this area; see (Vereshchagin and Vitányi, 2002). Kolmogorov's structure function is closely related to J. Rissanen's *minimum description length principle* for inductive inference. In its simplest guise, this says that the best theory for a given set of data is the theory that minimizes the description length of the theory plus the description length of the data given the theory. Thus, data is encoded by first encoding a theory (constituting the 'structural' part of the data) and then encoding the data using the properties of the data that are prescribed by the theory. Picking the theory minimizing the total description length leads to an automatic trade-off between complexity of the chosen theory and its goodness of fit on the data. This provides a practical and successful principle of inductive inference that may be viewed as a mathematical formalization of 'Occam's Razor'. But that is quite another story – we refer to (Grünwald, 2003) and (Rissanen, 1989) for details.

References

Chaitin, G. J. (1987), *Algorithmic Information Theory.* Cambridge University Press.

Cover, T. M. and Thomas J.A. (1991), *Elements of Information Theory.* Wiley Interscience, New York.

Gács, P. (1974), On the symmetry of algorithmic information. *Soviet Math. Dokl.*, 15:1477–1480, 1974. Correction, Ibid., 15:1480.

Gács, P., J. Tromp, and P. Vitányi (2001), Algorithmic Statistics. *IEEE Trans. Inform. Th.* 47(6): 2443–2463.

Grünwald, P.D. (2003), Manuscript, CWI.

Fisher, R. A. (1922), On the mathematical foundations of theoretical statistics, *Philosophical Transactions of the Royal Society of London, Ser. A*, 222: 309–368.

Kolmogorov, A.N. (1965), Three approaches to the quantitative definition of information. *Problems Inform. Transmission*, 1(1):1–7.

Kolmogorov, A.N. (1974), Talk at the Information Theory Symposium in Tallinn, Estonia, according to P. Gács and T. Cover who attended it.

Li, M. and P.M.B. Vitányi, (1997), *An Introduction to Kolmogorov Complexity and Its Applications.* Springer-Verlag, New York, revised and expanded second edition.

Merhav, N. and M. Feder (1998), Universal prediction, *IEEE Trans. Inform. Theory*, 44:6, 2124-2147.

Rissanen, J. (1989), *Stochastic Complexity in Statistical Inquiry.* World Scientific Publishing Company.

Rissanen, J. (2202), Kolmogorov's structure function for probability models. In: *Proc. IEEE Information Theory Workshop.* pp. 98–99, IEEE Press.

Shannon, C.E. (1948), The mathematical theory of communication. *Bell System Tech. J.*, 27:379–423, 623–656.

Shen', A.Kh. (1983), The concept of Kolmogorov (α, β)-stochasticity and its properties. *Soviet Math. Dokl.*, 28:295–299.

van Rooy, R. (2003), Quality and Quantity of Information Exchange. *Journal of Logic, Language and Information*, this volume.

Vereshchagin, N.K. and P.M.B. Vitányi (2002), Kolmogorov's structure functions and an application to the foundations of model selection, *Proc. 47th IEEE Symp. Found. Comput. Sci.*, 751–760.

Vitányi, P.M.B. and M. Li, (2000), Minimum Description Length Induction, Bayesianism, and Kolmogorov Complexity, *IEEE Trans. Inform. Theory*, IT-46:2, 446–464.

Vitányi, P (2002), Meaningful information. In: *Proc. 13th International Symposium on Algorithms and Computation (ISAAC)*, Vol. 2518 of *Lecture Notes in Computer Science*. Berlin, pp. 588–599, Springer Verlag.