6.441 Transmission of Information May 18, 2006

Lecture 24

Lecturer: Madhu Sudan *Scribe: Chung Chan*

1 Kolmogorov complexity

Shannon's notion of compressibility is closely tied to a probability distribution. However, the probability distribution of the source is often unknown to the encoder. Sometimes, we interested in compressibility of specific sequence. e.g. How compressible is the Bible? We have the Lempel-Ziv universal compression algorithm that can compress any string without knowledge of the underlying probability except the assumption that the strings comes from a stochastic source. But is it the best compression possible for each deterministic bit string? This notion of compressibility/complexity of a deterministic bit string has been studied by Solomonoff in 1964, Kolmogorov in 1966, Chaitin in 1967 and Levin.

Consider the following n-sequence

$$0100011011000001010011100101110111\cdots$$

Although it may appear random, it is the enumeration of all binary strings. A natural way to compress it is to represent it by the procedure that generates it: enumerate all strings in binary, and stop at the n-th bit. The compression achieved in bits is

$$|\text{compression length}| \leq 2\log n + O(1)$$

More generally, with the universal Turing machine, we can encode data to a computer program that generates it. The length of the smallest program that produces the bit string $\boldsymbol{x}$ is called the Kolmogorov complexity of $\boldsymbol{x}$.

Definition 1 (Kolmogorov complexity) *For every language $\mathcal{L}$, the Kolmogorov complexity of the bit string $\boldsymbol{x}$ with respect to $\mathcal{L}$ is*

$$K_{\mathcal{L}}(\boldsymbol{x}) = \min_{p:\mathcal{L}(p)=x} l(p)$$

where p is a program represented as a bit string, $\mathcal{L}(p)$ is the output of the program with respect to the language $\mathcal{L}$, and $l(p)$ is the length of the program, or more precisely, the point at which the execution halts.

But this notion of complexity depends on the particular language $\mathcal{L}$, which seems too specific to be useful. For example, we may have a language that prints the bible by a short program, say the ASCII string of "print the bible". The length of the program depends of the language so much that it does not seem to reflect our intuitive understanding of what compressibility is. Without fixing a particular language, however, the notion of complexity is ill-defined. Fortunately, we have the following theorem of the universal language which roughly says that the Kolmogorov complexity of $\boldsymbol{x}$ with respect to a universal language is well-defined up to a constant.

Theorem 2 (Universal language)

$$(\exists \textit{ universal language } \mathcal{U})(\forall \textit{ language } \mathcal{L})(\exists \textit{ finite constant } C_{\mathcal{L}})(\forall \textit{ bit string } \boldsymbol{x})$$
$$K_{\mathcal{U}}(\boldsymbol{x}) \leq K_{\mathcal{L}}(\boldsymbol{x}) + C_{\mathcal{L}}$$

Can we choose the best universal language $\mathcal{U}$ that minimizes $C_{\mathcal{L}}$ over all choices of $\mathcal{L}$? Such a minimum may not exist because $C_{\mathcal{L}}$, although finite, is unbounded.[1]

[1] Given any universal language $\mathcal{U}$ that one claims to be the best, we can find a finite bit string $\boldsymbol{x}$ that $\mathcal{U}$ compresses to more than 1 bit, and then give a universal language $\mathcal{U}'$ that compresses $\boldsymbol{x}$ to exactly 1 bit by storing $\boldsymbol{x}$ within the language manual.

To relate Kolmogorov complexity to Shannon's notion of compressibility, let us ask the following question: what is the probability model $P_{\mathcal{U}}$ under which the compression using the shortest program is good? Intuitively, sequences that can be generated by shorter programs should be more probable. i.e. $K_{\mathcal{U}}(\boldsymbol{x}) \approx \log \frac{1}{P_{\mathcal{U}}(\boldsymbol{x})}$ along the idea of entropy encoding. This can be satisfied with the following model for generating $\boldsymbol{X}$,

1. Fix a universal language $\mathcal{U}$.
2. Generate a random program p from an iid Bern(0.5) source.
3. Generate $\boldsymbol{X}$ as the output $\mathcal{U}(p)$.

The corresponding distribution $P_{\mathcal{U}}$ is called the universal probability, defined as follows,

Definition 3 (Universal probability)

$$P_{\mathcal{U}}(\boldsymbol{x}) = \sum_{p:\mathcal{U}(p)=\boldsymbol{x}} 2^{-l(p)}$$

1.1 Spectrum of research on data compression

Shannon's model assumes a known distribution, and easy optimal compression schemes are available. On the other hand, Kolmogorov model is robust under unknown distribution, but the compression, which involves searching for the shortest programs for strings, is incomputable. Fortunately, there are a variety of models between these two extremes.

Lempel Ziv Model assumes an finite-state Markov chains that may be unknown to the encoder. The compression algorithm is easy and is implemented in practice.

Keiffer-Yang uses grammars to compress strings. For example, a grammar may consists of the following rules with the associated probabilities:

$$\begin{array}{lll}
\text{sentence} \xrightarrow{.9} \text{subject,Verb,Object} & \text{subject} \xrightarrow{.9} \text{noun} & \\
\text{sentence} \xrightarrow{.1} \text{one word} & \text{subject} \xrightarrow{.1} \text{pronoun} & \\
\text{Verb} \xrightarrow{.2} \text{an} & \text{pronoun} \xrightarrow{.99} \text{I} & \text{object} \xrightarrow{.3} \text{me} \\
\vdots & \vdots & \vdots
\end{array}$$

Charikar, Sahai, Lehman etc. have come up with efficient grammars for polynomial-time compression algorithms.

Resource bounded Kolmogorov complexity $K_{\mathcal{U}}^{n^2}(\boldsymbol{x})$ is defined as the length of the smallest program p that produces $\boldsymbol{x}$ in time $l(\boldsymbol{x})^2$. The compression can be done within $2^{l(\boldsymbol{x})^2}$, basically by searching through all the possible programs that satisfies the resource constraint.

2 Summary of the course

Starting with the basic probability theory, we defined the entropy and mutual information as an intuitive measure of the uncertainty of a random variable and that of the information shared between two random variables. Then, we introduced the AEP, which comes in handy when we tackle large objects from small processes. In particular, we applied it in typical set source encoding and channel decoding. Random encoding is another important notion that simplified the error analysis when proving achievability results of source and channel coding. We then introduced the differential entropy as the appropriate measure of randomness of a continuous random variable, not in the absolute sense, but in the relative sense for the purpose of comparing to another random variable in the same coordinate system. Finally, we

introduced the network information theory for multiple access and broadcast channels, coding theory and Kolmogorov complexity. The need for information theory to address computational complexity is important for designing practical systems, such as channel codes with efficient encoding and decoding algorithms, and cryptographic systems that is computationally infeasible to break.

There are some topics that we wish to have covered. For example, the applications of information theory outside the communication settings such as Gambling and Stock Market, and the rate distortion theory.