2002-10-01

Bluetooth: An Emerging Technology for Wireless Personal Area Networks

Eric S. Hall
enrique.hall@gmail.com

Charles D. Knutson
knutson@cs.byu.edu

See next page for additional authors

Original Publication Citation
Charles D. Knutson, David K. Vawdrey, Eric S. Hall. "Bluetooth: An Emerging Technology for Wireless Personal Area Networks." IEEE Potentials Magazine, October/November, 22.

Bluetooth

...an emerging technology for Wireless Personal Area Networks

Charles D. Knutson, David K. Vawdrey and Eric S. Hall

Look under your desk or behind your computer. See that rat's nest of wires and cables? Almost every cable or wire that tethers us to our devices (or our devices to one another) is unnecessary. Bluetooth wireless technology promises to reduce the cabling chaos that afflicts us all.

Bluetooth is a recently developed technology that uses radio frequency (RF) transceivers to provide point-to-multipoint wireless connectivity within a personal space. Bluetooth was designed for both voice and data communication at low per-unit costs while consuming little power. To achieve the cost and power goals, Bluetooth limits connectivity to a sphere of about 10 meters (more power-hungry versions can stretch the effective range up to 100 meters) while providing a maximum data rate of 723 kbps.

The alternatives

Of course, Bluetooth is not the only short-range wireless game in town. Let's not forget Wi-Fi (802.11b wireless Ethernet). Wi-Fi operates in the same 2.4 GHz RF band as Bluetooth, although it uses a different spread spectrum approach to avoid interference (direct sequence, rather than frequency hopping). The two technologies can generally co-exist in the same frequency band. They, also, can complement one another in certain situations, particularly when both are present in the same device.

However, Wi-Fi and Bluetooth were not designed to solve the same problems. They are actually no more competitors than pickup trucks are with motorcycles. Their roles are quite different: Wi-Fi was designed as a wireless LAN solution while Bluetooth was designed primarily as a cable replacement technology for consumer electronic devices.

So what about that *other* cable replacement technology, IrDA? It still lives, and appropriately so. Past work by the Infrared Data Association delivered interoperable infrared solutions for personal wireless connectivity. Infrared communication is well suited for applications requiring quick discovery, short-range point-to-point ad hoc connectivity. Recent IrDA successes in the area of infrared financial messaging are particularly promising. Still, infrared is not particularly well suited for situations such as printing to a device next door (or anything that requires going through a wall for that matter, hot syncing from a distance, or using a wireless headset with a cell phone. Enter Bluetooth.

Early work

The effort to define and deploy Bluetooth was initiated by Ericsson, the cellular telephone manufacturer based in Sweden. Engineers at Ericsson envisioned the benefits of wirelessly connecting cell phones to other devices. For example, a laptop might use a cell phone as a modem to allow email and Internet access. The same cell phone could also connect to a wireless headset, providing hands-free communication capabilities to the wearer. The phone might also coordinate with Personal Information Management (PIM) data in a Personal Digital Assistant (PDA), synchronizing phone numbers and calendar items.

Limiting cost was a key factor in the creation of Bluetooth technology. The stated target price from the beginning was $5 for the transceiver and chipset. After months of hype, this price point is finally becoming a reality. By providing standard hardware and software functionality at an affordable price, Bluetooth is poised to enable a plethora of tether-free devices that can interoperate with one another, irrespective of original manufacturer.

The Bluetooth SIG

After the Bluetooth effort began at Ericsson in 1994, numerous high-tech companies quickly climbed on board. The Bluetooth Special Interest Group (SIG) was formed in 1998 by founding companies Ericsson, Intel, IBM, Nokia and Toshiba. This original group of "Promoters" was joined in 1999 by 3COM, Lucent, Microsoft and Motorola. Today Bluetooth SIG membership sits at around 3,000 member companies, representing a significant cross-section of the wireless and mobile computing industry.

The Bluetooth SIG is a trade association whose purpose is to maintain interoperability through strict qualification procedures and regular product testing. The SIG also supports several working groups involved in engineering, qualification, and marketing. Member companies obtain rights to use the Bluetooth brand and receive access to updated specifications.

Bluetooth profiles

A technology is only as relevant as the value it brings to the user. With that in mind, the Bluetooth SIG has defined a number of profiles describing specific uses for the technology from the user's perspective. These profiles are organized in a hierarchical fashion, permitting conceptual reuse between profiles (see Fig. 1). For example, all profiles take advantage of the Generic Access Profile, while only those profiles that deal with the movement of data objects utilize the Generic Object Exchange Profile.

New profiles are developed as new uses and needs arise. The goal of the profiles is to maximize interoperability between Bluetooth-enabled devices. The idea is to assure that for a given usage scenario, all similarly enabled devices will use the technology in the same way. Companies must not only certify that their devices implement established profiles, but that they implement those profiles in the standard way.

The following are examples of current Bluetooth profiles.

• *Modem*—This profile permits a cell phone to establish a Bluetooth connection to a laptop or PDA, allowing the remote device to use the phone as a modem. In practice, this means that a

0278-6648/02/$17.00 © 2002 IEEE

Bluetooth-enabled phone that implements the Modem Profile can provide a mobile user with Internet access so long as there is cellular coverage.

• *Headset*—This profile allows a headset to wirelessly connect to a peripheral device (such as a cell phone). This way a user may use a cellular headset in a truly tether-free fashion. It also implies that if a user has a device that is a combination PDA and cell phone, the cellular capabilities can be used simultaneously with other PDA functions.

• *Network Access Point*—For devices that wish to access local or wide- area networks, this profile permits a Bluetooth-enabled device to bridge to the IP-based infrastructure via a wireless access point.

• *Object Exchange*—This profile defines a mechanism for a data object exchange between peer devices. This is similar to the "point and shoot" model used by IrDA, but it also includes other sub-profiles for a more refined object exchange.

• *Synchronization*—A specialized example of the Object Exchange profile, Synchronization describes the mechanisms by which peer devices can coordinate and share data. This profile may include obvious synchronization scenarios (such as a PDA with a laptop); but, it may also include selective peer synchronization (such as a PDA with a cell phone or one cell phone with another).

To facilitate these profiles, a common hardware and software foundation has been established. The Bluetooth specification is an extensive document, spanning more than 1,000 pages and covering technical details from low-level hardware design up through the core protocol stack to profile-enabling session layer protocols (see Fig. 2). The following sections briefly describe the technical issues laid out in the Bluetooth specification.

Bluetooth hardware basics

Bluetooth is a short-range RF technology operating in the 2.4 GHz ISM (Industrial, Scientific, Medical) band. Since it uses radio frequency, Bluetooth has no direct line of sight restrictions and penetrates most physical barriers. Bluetooth divides the 2.4 GHz band into 79 channels of 1 MHz each, with a lower guard band of 2 MHz, and an upper guard band of 3.5 MHz.

By utilizing Frequency Hopping Spread Spectrum (FHSS) techniques, the Bluetooth radio hops rapidly in a pseudo-random pattern among the channels, lingering only 625 ms on any given channel frequency. Since Bluetooth supports 1, 3 and 5 slot packets for asynchronous data, the maximum time that can be spent transmitting on any given channel is 3.125 ms. By utilizing 5 slot packets, with no forward error correction (FEC), a maximum data rate of 723.2 kbps can be achieved.

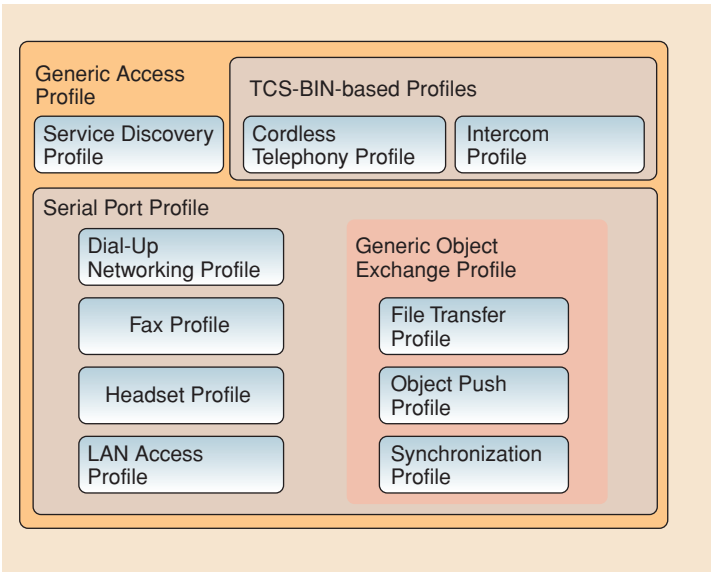

Fig. 1 Profiles

To facilitate power-sensitive devices, the Bluetooth radio supports three power classes. Class 1 limits maximum output power to 100 mW, class 2 allows 2.5 mW, and class 3 is the lowest power consumer, outputting a maximum 1 mW. Using the class 3 radio, a Bluetooth connection extends 10 meters, while the class 1 radio provides a transmission range greater than 100 meters.

Two link types

Separate modes are supported for voice and data transmission. Synchronous Connection-Oriented (SCO) links are circuit-switched connections between a "master" device and one "slave." SCO links are used exclusively for audio or other time-bound signals. In contrast, Asynchronous Connection-Less (ACL) links are used to form point-to-multipoint packet-switched connections between a master and multiple slaves. ACL links are used for data communication, and provide several differ-

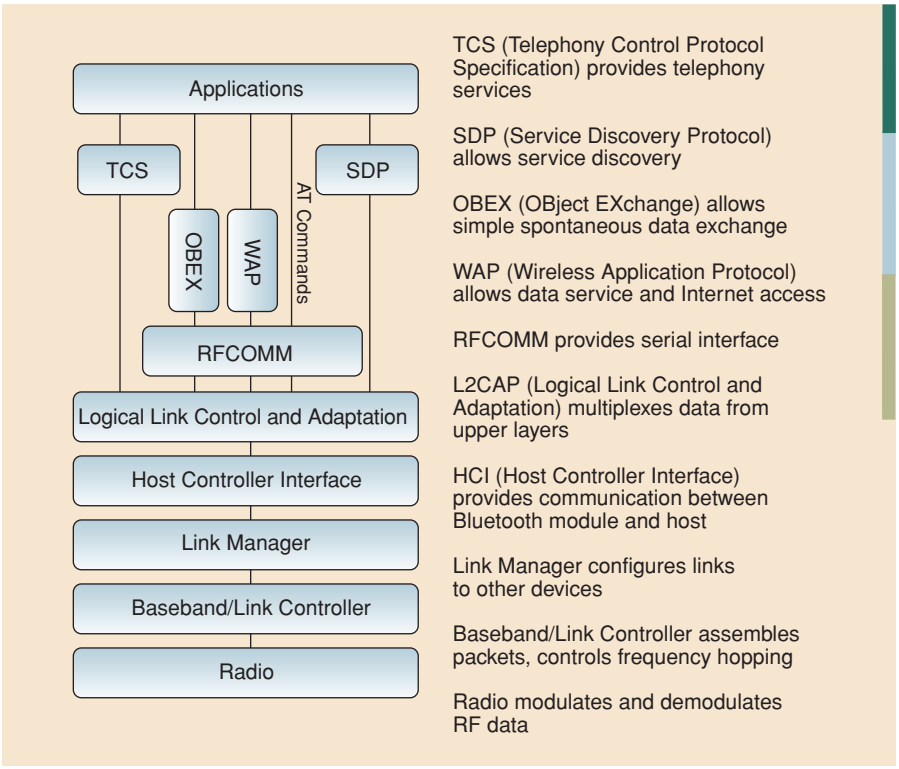

Fig. 2 Bluetooth Protocol Stack

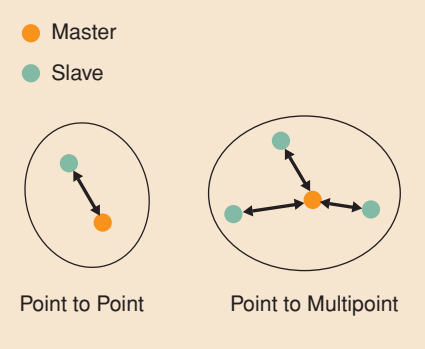

Fig. 3 Piconets

ent packet types with varying levels of Forward Error Correction (FEC). A Bluetooth master may connect with up to seven active slaves to form a personal area network called a "piconet."

Masters, slaves, piconets and scatternets

Bluetooth devices may play either master or slave roles, although specific connections are limited to a single master with one or more slaves. Piconets are initiated and formed by a master (see Fig. 3). The master establishes a hopping sequence (based on its clock and hardware address), performs inquiry to discovery potential slaves, and pages appropriate devices to invite them to join the piconet.

The master also controls the flow of data transmission. The master of a piconet always transmits in odd-numbered time slots (see Fig. 4). All active slaves listen long enough to determine if a packet is intended for them. If a slave receives a packet, it transmits on the subsequent even-numbered slot. In this way, Bluetooth employs a form of time division multiplexing in addition to the frequency division multiplexing provided by frequency hopping.

To facilitate higher data rates, a master or a slave may transmit multi-slot packets of 3 or 5 slots (see Fig. 5). When the receiving device detects the multi-slot packet, it continues to listen through both odd and even slots. Other devices not involved in the communication continue their frequency hopping as usual. The other two devices join them in the normal flow of hopping when they have completed their multi-slot packet transmission.

Scatternets are a logical extension of the piconet concept. They permit a device to participate in multiple piconets simultaneously (see Fig. 6). In doing so, such devices can potentially serve as bridges between two piconets. The sharing device may either be a master in one piconet and a slave in another, or it can be a slave in two piconets. Since the hopping sequence of a piconet is determined by the clock and the hardware address of the master, it is not possible for a device to be the master of two piconets at the same time.

Devices may participate as a connected member of a piconet at a number of levels. The most obvious example is the active mode, where a slave actively participates in the piconet by listening to all the odd-numbered packets to see if one is addressed to it.

Three power-saving modes are also available. In park mode, a slave remains synchronized, but does not actively participate by listening or responding. In hold mode, a slave ceases to participate in a piconet for an agreed upon period of time, after it which it rejoins the piconet as an active slave. In sniff mode, the master sends to a specific slave less frequently, reducing the active listening requirements for the slave.

All of the capabilities described are provided in the Bluetooth baseband. Between the Bluetooth baseband controller and the host device is a layer called the Host Controller Interface (HCI). The HCI abstracts all of the capabilities of the baseband to the Bluetooth protocol stack on the host side.

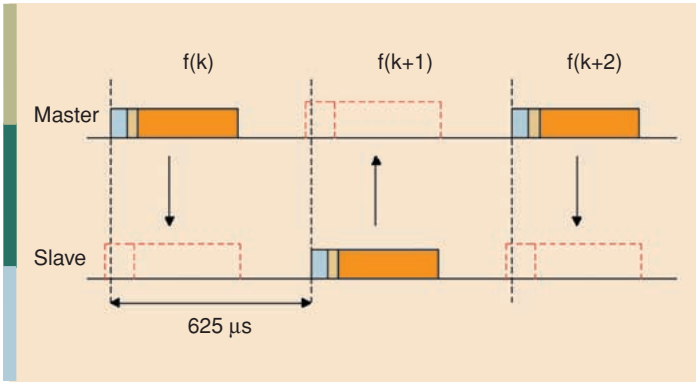

Fig. 4 Single time slots

Bluetooth protocol stack

A protocol stack is a layered set of communication software modules. Applications communicate by passing data and connection information to the top of the stack. There it is handed down through all stack layers (each performing some necessary operation, and typically appending a separate header to the payload) until it reaches the physical or hardware layer. At this point, it is transmitted to the remote device.

The physical layer on the receiving side detects the signal, reassembles the transmitted packet and passes it up the stack where analogous operations are performed and headers are removed until the payload eventually reaches the application.

King Harald

While pursuing their goal of a standardized wireless cable replacement technology, engineers at Ericsson took inspiration from the legendary Harald Blåtand, a 10th century King of Denmark. Among his many achievements, King Harald is most famous for uniting Denmark and Norway and for bringing Christianity to Scandinavia. The name Blåtand is probably derived from the words "ble" (blue) meaning dark or tanned, and 'tan' meaning a great man. The concatenated word "Blatand" translates loosely as "Bluetooth" in modern English. However, it almost certainly had nothing to do originally with either teeth or the color blue (current popular urban legends notwithstanding).

—CDK, DKV & ESH

The Bluetooth protocol stack resides on the host and provides interfaces for a variety of software applications. The core connectivity layer is the Logical Link Control and Adaptation Protocol (L2CAP). This layer interfaces with the HCI and presents an interface to upper layers for ACL data transfer. L2CAP also performs protocol multiplexing, permitting communication channels for multiple applications via the Bluetooth protocol stack. Finally, L2CAP performs segmentation and reassembly, as well as managing Quality of Service (QoS) features.

One of the most important layers above L2CAP is RFCOMM, which furnishes serial port emulation. It provides up to 60 simultaneous connections over the same link. Most higher-layer-data-transfer protocols are built on RFCOMM, including the Object Exchange (OBEX) protocol.

OBEX was actually created by the Infrared Data Association for use in point-to-point object exchanges. As an

example, when you use the infrared beaming feature of PDAs, the protocol used for that exchange is normally OBEX. The Bluetooth SIG determined that there were advantages in using a protocol like OBEX. First of all, an application written for OBEX can function correctly independent of the underlying transport used to actually move data. In other words, an OBEX-enabled application for an embedded platform could be ported naturally between devices enabled with IrDA and others enabled with Bluetooth.

Beyond this obvious advantage, devices that support *both* IrDA and Bluetooth can be designed to synergistically use both transports. Such devices might select one transport or the other, depending on availability. In a more creative scenario, an application could dynamically switch between the two transports in the middle of a transaction to handle changing conditions.

© PHOTO DISC COMPOSITE: MKC

Other layers above RFCOMM include WAP-enabled TCP/IP functionality and AT commands. Because of RFCOMM's simple abstraction of serial data communication, a multitude of high-level protocols can be implemented above the Bluetooth protocol stack.

Since multiple services can be supported on a Bluetooth protocol stack, a mechanism is required to allow services to register locally and to be discovered remotely. The Service Discovery Protocol (SDP) is an extensible client-server system that is used for locating remote services between Bluetooth devices. Services are identified within SDP by class and attribute, permitting queries by remote devices for services with specific characteristics. In addition, SDP permits service browsing: a device can systematically explore all the services registered with the remote device. SDP uses L2CAP for remote communication.

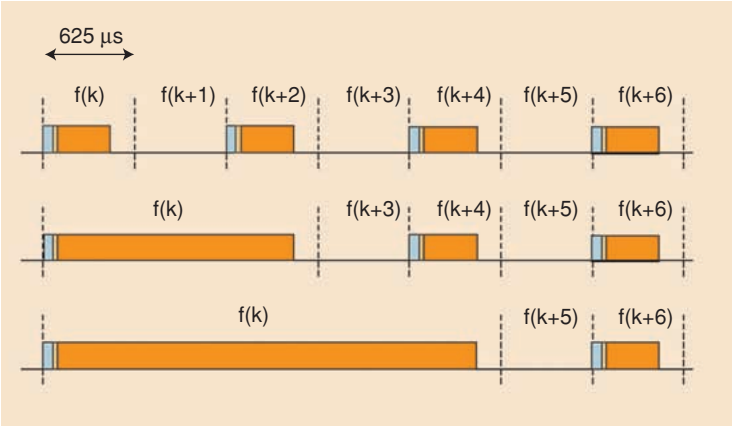

Fig. 5 Multi-slot packets

Encryption and security

Since wireless links are inherently insecure, Bluetooth implements a number of measures to enhance security. In the baseband, SAFER+ encryption algorithms are employed to protect data. However, secret key information is never broadcast over the air. It must be either entered by hand or built in by the manufacturers. As a result, this approach does not provide dynamic encryption between arbitrary Bluetooth devices.

Using the existing capabilities of the Bluetooth system, three modes of security are possible. Mode 1 does not utilize any of the available security features and implies a link that is not secure. Mode 2 utilizes security features initiated by applications and services. Lastly, Mode 3 involves security measures that are instituted any time a new connection is established. Additional security features may be implemented at the application layer.

A wireless future

Bluetooth is helping to usher in a new era of short-range wireless connectivity. As transceiver costs decline and handheld computing devices proliferate, Bluetooth is well positioned to bring significant value to a wide variety of users. New Bluetooth SIG working groups will continue to form to meet emerging user needs with new profiles. Look for Bluetooth to co-exist harmoniously with other short-range wireless technologies such as IrDA and 802.11b. Look also for increasingly synergistic uses for these complementary wireless technologies within the same device. The rat's nest under your desk won't vanish overnight, but because of short-range wireless technologies such as Bluetooth, we're moving in the right direction.

About the authors

Dr. Charles D. Knutson is an Assistant Professor of Computer Science at Brigham Young University (BYU) and Director of the BYU Mobile Computing Laboratory. He holds a Ph.D. in Computer Science from Oregon State University and B.S. and M.S. degrees in Computer Science from BYU.

Eric Hall received the B.S. degree in Computer Engineering from Brigham Young University in 2001, and is currently pursuing an M.S. degree in Computer Science. He is a research assistant in the Mobile Computing Laboratory at BYU.

David Vawdrey received the B.S. degree in Computer Engineering from Brigham Young University and is pursuing an M.S. degree in Computer Science. He is a research assistant in the Mobile Computing Laboratory at BYU.

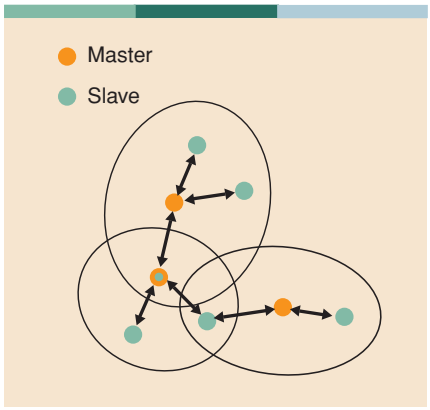

Fig. 6 Scatternets

Information, uncertainty and randomness

algorithmic perspectives

Ashish Pandharipande

"Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin." —John von Neumann

What is information? How can it be measured? What is randomness? When can a given string of numbers be classified as random?

Consider the following strings, or sequences, of outcomes obtained by tossing a fair coin forty times with each outcome resulting in a Heads (H) or Tails (T):

(a) HTHTHTHTHTHTHTHTHTH-THTHTHTHTHTHTHTHTHTHT

(b) HTTHHHTHHHHTTHTTTHTH-HTTTTHTHTTHHTTHTHTHH

Sequence (a) is clearly simpler; it consists of patterns of HT repeated 20 times. Sequence (b) however exhibits no such simple pattern. Tossing a coin 40 times can result in 2^{40} distinct strings, each of which can occur with an equal probability. Thus, obtaining pattern (a) should come as no greater surprise than obtaining pattern (b). If the provenance of strings in a probabilistic framework were to be the sole criterion to determining randomness, then both strings would have to be considered random. Nevertheless, we would like to capture our intuitive notion that string (b) is somehow less ordered, and hence more "random," than string (a). In keeping with this notion, we will shift our emphasis from the generating process of strings and focus, instead, on individual outcomes. We will focus on the algorithmic notions of information and randomness.

Classical probabilistic notions

The field of information theory has grown considerably since Claude Shannon's seminal work. Though Shannon's work on information theory was motivated by problems of communication theory, the realm of information theory is not limited to communications. A fundamental question that one seeks in information theory is "How does one measure an amount of information?" Certain quantities in information theory, such as entropy and mutual information, arise as answers to these fundamental questions.

According to Shannon's theory, information is a measure of one's freedom of choice when one selects a message. If one is faced with an equal choice between two alternative messages, the information associated with this situation is unity. The concept of information here applies not to the individual messages; it applies rather to the situation as a whole, the unit information indicating that one has an amount of freedom of choice, in selecting a message, which is regarded as unit amount.

To be precise, let E be an event occurring with probability p_E. Given that event E has occurred, we say that

$I(E) = \log (1/p_E)$

units of information have been received. Now consider a source emitting a sequence of symbols from a finite fixed set $S = \{s_1, s_2, \ldots, s_N\}$. Successive symbols from the source are statistically independent and selected according to some fixed probability law—probability with which symbol s_i occurs is p_i, $i = 1, 2, \ldots, N$.

We have just seen that if symbol s_i occurs, then we obtain an amount of information equal to $I(s_i) = \log (1/p_i)$ bits. The probability that s_i occurs is p_i, and the average amount of information per symbol, called entropy, is then

$H(S) = \hat{A}_i\ p_i\ I(s_i) = \hat{A}_i\ p_i \log (1/p_i)$ bits/symbol.

$I(s_i)$ can be interpreted as the information needed to make the occurrence of symbol s_i certain. $H(S)$ can be interpreted as the average amount of uncertainty which an observer has before his inspection of the output of the source. Entropy is, thus, a probabilistic measure of uncertainty or ignorance; information is a measure of the reduction in that uncertainty.

Algorithmic notions

One way to measure the information content of a given string is to determine the size of the smallest string or code that can be reproduced by some computer. Here a computer takes as input a program, or string of instructions, and produces a desired string as output. If it is possible to reduce redundancy by compressing the string, then a shorter sequence can serve as a code to reproduce the whole. In a truly random string, all the redundancy has been squeezed out and no further compression is possible. Accordingly, randomness implies the absence of any compressing possibility: the string has maximum information content.

There are many ways to compress information; an important one consists in detecting patterns. Consider a situation wherein we need to convey a very long table of trigonometric functions to a person who is visiting a planet in another galaxy. We could simply translate the numbers into an appropriate code, say in binary form, and transmit them directly. However even the most modest tables would have thousands of digits, making the cost of such transmissions prohibitive.

A much cheaper way to convey the same information would be to transmit instructions for calculating the tables from the underlying trigonometric identities, such as Euler's equation $e^{ix} = \cos x + i \sin x$. Such a message would be relatively brief. It also would still have all the information contained in even the largest tables.

This aforementioned method however fails to be adequate for empirical data. For instance, if we were asked to transmit game scores and match statistics of all four Grand Slam Tennis tournaments played to date, the possibility of compression in this information would be practically nil. These ideas lead us to a definition of randomness: information embodied in a random series of numbers that cannot be compressed, or reduced to a more compact form. If a string does have any redundancy, then a shorter sequence obtained by removing this redundancy can serve to reproduce the original string.

0278-6648/02/$17.00 © 2002 IEEE

The classical theory of information is based on Shannon's concept of entropy. Entropy is a measure of ignorance concerning which possibility is held in a set endowed with an *apriori* probability distribution. However, the algorithmic information theory adopts as a primary concept the descriptional complexity of an individual object. It dispenses with probability distributions.

The classical definition of randomness in probability theory allows one to speak of a process, such as the tossing of a coin, as being random. It does not allow one to call a particular outcome or string or sequence of outcomes, like obtaining twenty heads in a row with forty tosses of a fair coin, random (except in a heuristic sense).

In contrast, the algorithmic definition of probability makes no recourse to probability. Instead, it depends only on the availability of a procedure for computing the string. Moreover, the algorithmic definition is independent of the provenance of the string. It looks at the string itself as merely a succession of digits. The algorithmic complexity of an object is a measure of the difficulty of specifying that object; it focuses the attention on the individual, allowing one to formalize the intuition of randomness. An algorithmically random string cannot be produced from a description shorter than itself.

The main applications of algorithmic information theory are twofold. First, it provides a mathematical definition of what it means for a string to be patternless, disordered or random. Indeed, most strings are algorithmically irreducible and therefore random.

Second, and of great consequence, algorithmic information theory casts an entirely new light on Gödel's incompleteness theorem. This theorem states that there exist true statements within consistent mathematical systems that are unprovable using only the axioms of that system. Algorithmic information theory does this by placing information-theoretic limits on the power of any formal axiomatic theory.

In this article, we shall not delve deep into the second application. The interested reader can look up the references given at the end, in particular, the articles "Randomness and Mathematical Proof" and "Gödel's Theorem and Information" in Chaitin's book.

Algorithmic information theory has other interesting applications in characterizing complexity of systems and physical phenomena, in analyzing information content of genomes, DNA, and in life theory and mathematical biology.

Algorithmic complexity and randomness

We now present some intuitive notions of the concepts involved in algorithmic information theory. We start by examining the infinite continuum of numbers between zero and one when written in decimal expansion notation. Some of the numbers in this continuum, like .00000000… and .01010101…, exhibit a certain order and are simple. On the other hand the number .17853420942116… looks rather disordered. The question then arises as to how one characterizes the distinction between these numbers more precisely? To answer this question, we first address the question: How can we compute these and other numbers?

In his original work, Alan Turing distinguished between *computable* and *non-computable* numbers. To make this distinction, Turing considered writing a program, or an algorithm, for a computer that would compute various numbers.

For example, consider the number .69314718…. At first glance, this number looks random and lacking any discernible order. But what if we recognize this number as the first few digits of $\log_e 2$, expressed in decimal form? Then an algorithm that computes this number is "Compute $\log_e 2$ and print the result"—a simple program. Likewise, the algorithm for .10101010 would state "Print 10 four times." These are examples of computable numbers because there is a simple program that gives us the numbers even if the numbers are infinitely long, like the decimal expansion of $\log_e 2$.

However, for the non-computable numbers, the only program that we have is to explicitly specify the number itself within the program. For example, the only program that would compute the number .17853420942116…, where "…" here means say a million digits, is "Print .17853420942116…", with "…" specifying all the million digits. That is an enormous increase in program length.

Turing's ideas allow us to characterize different numbers by the length of the program required to compute them. It is possible to write a relatively short program for computable numbers, even if they are infinitely long. For the non-computable, random numbers, the only algorithm that can describe the number is about as long as the number.

This distinction provided a basis for the definition of random numbers, put forth by Kolmogorov and Chaitin. That is, random numbers are those that require a computational program as long as the numbers, themselves. This definition was also implicit in the work on simplicity of scientific theories by Solomonoff.

Kolmogorov, Solomonoff and Chaitin independently advanced the idea that the complexity of a string of data can be defined as a measure of the length of the shortest binary program for computing the string. Thus, the complexity is the minimum description length. This definition of complexity turns out to be universal, that is, machine independent.

Minimal program

We now introduce the notion of *minimal program*. Any particular number can be computed by an infinite number of programs. For example, the number 2397 can be obtained from the programs "Add 1 to 2396," or "Divide 7200 by 3, and subtract 3 from the result," or "Multiply 51 by 47," or an infinite number of other programs. However, the minimal program is of special interest. That is the shortest program once we have encrypted it as a string of integers (or bits). It is easy to see that the string of integers representing the minimal program must be random whether, or not, the series it generates is random.

Let P be a minimal program for the series of digits, denoted S? If P is not random, then by definition, there must be another smaller program, say Q, that will generate S. But this would be in contradiction to the assumption that P was a minimal program. Hence, a minimal program has to be random.

The idea of a minimal program is closely related to algorithmic complexity. The complexity of a series of digits is the number of bits that must be put into a computing machine to obtain the original series as the output. The complexity is therefore equal to the size in bits of the minimal program of the series.

More precisely

We now make the intuitive ideas of complexity discussed so far more precise. Consider the following natural ordering of binary strings:

f, 0, 1, 00, 01, 10, 11, 000, 001, 010, 011, 100, 101, 110, 111, 0000, ...

with f denoting the null string. The *n*th binary string in this order is mapped to the natural number *n*, *n* = 0, 1, 2, ...

Denote the length of the string *s* to be *l*(*s*). Then considering *s* to be a natural number defined by the mapping, we have $l(s) = \hat{I}\log_2(s+1)^{\circ}$, where $\hat{I}x^{\circ}$ is the greatest integer less than or equal to *x*. The algorithmic complexity of a string *s*, $I_C(s)$, with respect to a computer *C* is defined to be

$I_C(s) = \min_{C(p)=s} l(p)$,

the length of the shortest program *p* that makes the computer *C* output *s*. If there is no program that can make *C* output *s*, then $I_C(s)$ is defined to be infinite.

Measuring randomness

Thus far, we have considered the notion of complexity to define randomness. The ideas of complexity can also be used to measure randomness. Given several *n*-digit sequences, it is theoretically possible to identify all those of complexity *n*-1, *n*-10, *n*-100, and so forth. We can, thereby, rank the sequences in decreasing order of randomness.

It is however not possible to set a particular numerical value in order to judge what degree of randomness actually constitutes randomness. The value ought to be set low enough so that numbers with obviously random properties are not excluded and high enough so that numbers with conspicuous patterns are disqualified. This fuzziness is reflected in the qualified statement that the complexity of a random sequence is approximately equal to the length of the sequence.

> More than anything else, mathematics is a method.
> -Morris Kline
> [24 Aug. 1979]

Most strings with *n* digits are random ones and strings having a nonrandom frequency distribution are the exception. Of all the possible *n*-digit binary numbers, there is only one that consists entirely of 0s and only one that comprises all 1s. All the rest are less orderly and the great majority qualify as random.

To choose an arbitrary limit, we can calculate the fraction of all *n*-digit binary numbers that have a complexity of less than *n*-*k*, for a given integer *k*. There are at most 2^i distinct (binary) programs of length *i*, with *i* = 1, 2, …, *n*-*k*-1. Hence, there are at most $(2^1 + 2^2 + \ldots + 2^{n-k-1}) = 2^{n-k} - 2$, or less than 2^{n-k}, programs that generate strings of length *n* having complexity less than *n*-*k*. So, there are fewer than 2^{n-k} programs of size less than *n*-*k*.

Since each of these programs can specify no more than one series of digits, fewer than 2^{n-k} of the 2^n numbers have a complexity less than *n*-*k*. Thus, the fraction of strings that are not *k-random* is less than $2^{n-k} / 2^n = 1/2^k$. With *k* = 10, for instance, it follows only about one series in a 1000 is not random, and can be compressed into a computer program more than 10 digits smaller than itself.

A random binary sequence x_n of length *n* may now be defined to be of maximal or near-maximal complexity: if its complexity $I(x_n)$ is not much less than *n*. On similar lines, an infinite binary sequence *x* may be defined to be random if all its initial subsequences x_n are random finite binary sequences. The infinite sequence *x* is random, if and only if, there exists a *c* such that for all positive integers *n*, the algorithmic information content of a length *n* subsequence of the string *x* is bounded from below by *n*-*c*. Similarly, a real number will be called random if the base 2 expansion of its fractional part is a random infinite binary sequence.

We can easily show that a specified given string is not random; one only needs to find a program that will generate the string and that is itself substantially smaller than the length of the given string. The program does not need to be a minimal program for the string; it only needs to be sufficiently small.

To demonstrate, however, that a given string is random is impossible. The difficulty is associated with attempting to establish a lower bound on its complexity, as explained earlier, and is in fact a natural manifestation of Gödel's incompleteness theorem.

Summary

To summarize, randomness of a string of numbers can be understood through three viewpoints:

1) A string is random if each number in the string is generated by some mechanism in an unpredictable manner. It is the disorder of the generating process that results in the randomness.

2) A string is random because it is completely unexpected; its entropy is maximal.

3) A string is random because no prescribed program of shorter length can generate its successive digits. Accordingly, randomness implies the absence of any compression possibility. Thus, the string has maximum information content. Since it is maximally complex in an algorithmic sense, the string can only be reproduced by explicitly specifying the string itself.

We conclude with a remark made by Kolmogorov: "Any attempt to detect a highly developed extraterrestrial civilization by trying to intercept a message sent out for the same or a similar civilization is apparently doomed to failure. A highly developed civilization probably knows how to encode its messages in a very economical way. That means that its messages have a complexity per bit (that is, the complexity divided by the length of the message) and these messages are therefore practically indistinguishable from random sequences of bits."

Read more about it

• E. J. Beltrami, *What is Random?: Chance and Order in Mathematics and Life,* Copernicus Books, 1999.

• G. J. Chaitin, *Algorithmic Information Theory*, Cambridge University Press, 1992.

• H. R. Pagels, *The Dreams of Reason: The Computer and the Rise of the Sciences of Complexity,* Simon and Schuster, 1988.

• C. E. Shannon, W. Weaver, *The Mathematical Theory of Communication*, University of Illinois Press, Urbana and Chicago, Illinois, 1963.

• A. N. Shiryayev, *Selected Works of A.N. Kolmogorov: Information Theory and the Theory of Algorithms,* Volume 3, Kluwer Academic Publishers, 1993.

About the author

Ashish Pandharipande is currently pursuing his doctoral degree in Electrical and Computer Engineering at the University of Iowa. He received his Masters degrees in Mathematics in 2001, and Electrical and Computer Engineering in 2000, both from the University of Iowa, and his Bachelors degree in Electronics and Communication Engineering in 1998 from the College of Engineering, Osmania University, India.

His research interests are in Multicarrier and Wireless Communi-cations, Multirate Signal Processing, and Signal Processing for Communications. Other interests include Discrete Mathematics, Cryptography and Graph Theory.