The following is a slightly shortened version of my article in Cryptologia*, Vol. 21, No. 4 (1997).*

Cryptography As a Teaching Tool

Cryptography has a tremendous potential to enrich math education. In the first place, it puts mathematics in a dramatic setting. Children are fascinated by intrigue and adventure. More is at stake than a grade on a test: if you make a mistake, your agent will be betrayed.

In the second place, cryptography provides a natural way to get students to discover certain key mathematical concepts and techniques on their own. Too often math teachers present everything on a silver platter, thereby depriving the children of the joy of discovery. In contrast, if after many hours the youngsters finally develop a method to break a cryptosystem, then they will be more likely to appreciate the power and beauty of the mathematics that they have uncovered. Later I shall describe cryptosystems that the children can break if they rediscover such fundamental techniques of classical mathematics as the Euclidean algorithm and Gaussian elimination.

In the third place, a central theme in cryptography is what we do *not* know or can*not* do. The security of a cryptosystem often rests on our inability to efficiently solve a problem in algebra, number theory, or combinatorics. Thus, cryptography provides a way to counterbalance the impression that students often have that with the right formula and a good computer any math problem can be quickly solved.

Mathematics is usually taught as if it were a closed book. Other areas of science are associated in children's minds with excitement and mystery. Why did the dinosaurs die out? How big is the Universe? M. R. Fellows [1] has observed that in mathematics as well, the frontiers of knowledge can and should be put within reach of young students.

Finally, cryptography provides an excellent opportunity for interdisciplinary projects. The first example in the next section shows how this can be done in the middle or even primary grades.

Elementary Examples

Example 1. Caesar and Vigenère. First we describe the Julius Caesar encryption by alphabet shifts. We start by choosing a fixed integer, such as 5. To encipher a message, each letter is replaced by the letter 5 places down the alphabet. To encrypt V, W, X, Y, or Z, we return to the beginning of the alphabet. For instance, the message ``YES'' gets transformed to ``DJX.'' The mathematical concept here is addition in a cyclic group. Schematic wheels provide a vivid way to illustrate this simple but important idea. In fact, actual wheels with gears were used at various times in history (see [4]) to implement this type of alphabet shift (and the closely related Vigenère shift that will be described below).

Perhaps the children can figure out a way to break the cipher on their own; otherwise, *frequency analysis* can be suggested to them. That is, we first ask them to guess which letter appears most frequently in English (most will say either `e' or `a'). Then we have each student choose a paragraph in a book, count the a's, e's, etc., and in this way decide whose conjecture is correct. The students might want to investigate how long a text one needs in order to have reasonable confidence that `e' occurs the most often. What strategies could one use for a short text to which the assumption about the frequency of `e' does not apply? What are the second and third most frequently occurring letters?

Spanish examples can be used even if most of the children do not know Spanish. It is a good idea to choose cognates and phrases that are familiar from the movies (like `hasta la vista'), so that the students will be able to figure out the meaning after deciphering. Before doing Spanish examples, the kids might find it interesting to compile frequency statistics from paragraphs in Spanish-language books.

Next, we introduce Vigenère ciphers, which work like the Caesar method, except that an entire block of 2, 3, 4, or 5 letters is shifted by a *key-word*. For example, if the key-word is ``dog," consisting of the 4th, 15th, and 7th letters of the alphabet, then the first letter of the message is shifted by 4, the second letter is shifted by 15, the third by 7, the fourth by 4 (here we return to the beginning of the key-word), the fifth by 15, and so on.

We encourage the children to figure out for themselves how to cryptanalyze Vigenère ciphers. Sometimes they will come up with the idea of guessing the length of the key-word (say, 3), and then applying frequency analysis to each third letter. Some of the same questions as before can be asked (how long a text does one need for the frequency analysis to work reliably?), along with some new ones (what happens if our guess for the key-word length is wrong?).

Along with mathematical concepts -- functions, inverse functions, cyclic groups, modular addition -- this topic introduces the students to ideas from other fields, such as statistics and languages. In addition, it provides a natural lead-in to a discussion of *one-time pads*, which are Vigenère ciphers of ``infinite" key-word length and are the only cryptosystem with perfect security.

In conjunction with a course I teach for math education majors, my students and I visit several 6th grade inner-city math classes in Seattle each week. One of the most successful enrichment topics that we present has been the Caesar and Vigenère ciphers. The examples I use (both English and Spanish) are available electronically.

Before giving the next example, I want to define the term ``Kid Krypto" that Fellows and I introduced in [3]:

Definition. *Kid Krypto* is the development of cryptographic ideas that are accessible and appealing (and moderately secure) to those who do not have university-level mathematical training.

Example 2. Information-Hiding Protocol. Suppose that a teacher wants to know the average amount of time her students spend on homework each week. She suspects that if she simply asks them, they will not give entirely truthful answers. The lazy ones might inflate their answers; and the diligent ones might understate their numbers so that their friends will not think that they are ``nerds." So instead she uses an information-hiding protocol that reveals the average but allows each child to keep his or her number of hours secret. She goes around the class in some order; suppose that the children are named Abdul, Busiso, Conchita, Dilia, ..., Zeineb. Abdul chooses a secret integer, adds his number of hours to it, and whispers the sum to Busiso. Busiso adds his number of hours to the number he heard from Abdul, and whispers the new sum to Conchita. Conchita adds her number of hours, and passes the sum to Dilia, and so on. Finally, Zeineb adds her number of hours and returns the sum to Abdul. Abdul subtracts his secret number and tells everyone the result. The teacher divides this total by the number of children, thereby determining the average time on homework.

If students collaborate, then clearly privacy is compromised. For instance, if Abdul and Conchita get together, they can determine how much time Busiso spends on homework. However, in my experience children enjoy playing by the rules, and do not ``cheat."

More Complicated Examples

Example 3. Kid-RSA. The following cryptosystem can be introduced among secondary school students who have learned how to reduce numbers modulo a positive integer n and how to convert numbers from one base to another. In particular, they must know how to work with blocks of letters regarded as integers to the base 26, where each letter is a digit (that is, A=0, B=1,..., Z=25).

To set up the system, a student (Alice) chooses any two integers a and b, sets $M=ab-1$, then chooses two more integers a' and b', and finally sets $e=a'M+a$, $d=b'M+b$, $n=(ed-1)/M= a'b'M+ab'+a'b+1$. Her public key is (n,e), and her private key is d. (The letter ``e" was chosen to signify ``encryption" and ``d" to signify ``decryption.") To send Alice a plaintext m, one uses the map $c=em \pmod n$; Alice deciphers the ciphertext by multiplying by d modulo n. Note that the decryption operation recovers the plaintext, because $dc=dem=m \pmod n$.

As a side excursion, one can show how to make digital signatures. This works as follows. Suppose that Bob wants to sign a message to Alice with the letters ``BOB.'' Suppose that his public key is (n',e'), and his private key is d'. Let s be the integer corresponding to his signature: $s = (\mathrm{BOB})_{26} = (1\ 26\text{^}2 + 14\ 26 + 1)_{10} = (1041)_{10}$. Bob first computes the least non-negative residue of $d's$ modulo n', then multiplies this number by e modulo n, and sends the result to Alice. Alice takes this number, multiplies it by d modulo n and then by e' modulo n'. It is not hard to see that she obtains $s = (\mathrm{BOB})_{26}$. Alice then knows that no one other than Bob could have sent her this signature, because no one other than Bob knows the value d' that is ``undone'' by e' modulo n'.

An interesting question is whether the youngsters will be able to break Kid-RSA. If they can find an efficient way to compute an integer d such that $de \equiv 1 \pmod{n}$ (not necessarily the same d that Alice has as her private key), then they can break the system, since multiplication by d modulo n inverts the encryption function, as we saw above. There is an efficient algorithm to find such a d that goes back to Euclid (see, for example, [7]), but students who have never studied number theory are not likely to know this algorithm.

Here is a question to which I do not know the answer: Can one prove that the ability to crack this cryptosystem (for any choice of a,b,a',b') implies the ability to solve the equation $xr+ys=1$ for any two relatively prime integers r and s? Could there be a way to crack Kid-RSA without essentially rediscovering a version of the Euclidean algorithm?

Example 4. Perfect Code Public Key System. Before introducing this cryptosystem, which is suitable for middle and high school students, we need some basic definitions from graph theory.

Definition. A *graph* is a collection of big dots called *vertices*, some of which are connected by lines called *edges*. The *neighborhood* of a vertex consists of the vertex itself and all vertices that are joined to it by an edge. A *perfect code* in a graph is a subset of vertices such that every vertex is in the neighborhood of one and only one vertex in the subset. Not all graphs have perfect codes, but all of the ones we shall use will have one or more perfect codes.

For example, consider the edge-graph of a cube, which consists of eight vertices and twelve edges. Any pair of opposite vertices will be a perfect code in this graph. (The term ``perfect code'' comes not from cryptography but from the theory of error-correcting codes. In the example of the edge-graph of a cube, the perfect code is a one-error-correcting Hamming code. Namely, take the points (x,y,z) with coordinates 0 or 1 to be the vertices of the cube and the points (0,0,0) and (1,1,1) to be the perfect code.)

Even if we know in advance that a given graph has a perfect code, it might be very difficult to find it. (Note: The problem of finding a perfect code in a graph is *NP-hard*. This means that if we had a fast method that was guaranteed to work in all cases, then there would also be a fast method for solving a vast class of problems, such as the Traveling Salesrep Problem and factorization of large integers. Moreover, the fundamental ``P not equal to NP'' conjecture of computer science would be false -- a consequence that would be as traumatic for computer scientists as a disproof of the Riemann Hypothesis would be for mathematicians. For this reason it is thought that there is no fast algorithm for the perfect code problem.)

However, there is an easy ``one-way'' construction of a graph with a perfect code. This means that, starting with a subset of vertices, it is possible to construct a graph around it that will have the subset as its perfect code. Suppose that you are given a sheet with a bunch of vertices (but no edges) already drawn. The vertices are numbered for easy reference.

STEP 1. First choose your perfect code **C**, consisting of only some of the vertices. Record the numbers of the vertices you have chosen, and *keep that list secret*.

STEP 2. Draw edges from the vertices in **C** to the other vertices, so that every vertex is connected to exactly one of the vertices in **C**. In other words, each vertex in **C** is the center of a ``star.'' The vertices not in **C** are the ``outer points'' of the stars.

STEP 3. Draw a bunch of additional edges connecting outer vertices of stars to each other. You may connect an outer vertex of a star to outer vertices of the same or different stars. *But do not draw any new edges from the center of any of the stars*. You should draw enough edges to ``disguise" the stars. That is, it should be very hard to guess where the centers of your original stars are located.

Suppose the letter *A* stands for someone who wants to be able to receive secret messages. *A* comes up with a so-called ``public key" that anyone else (denoted by the letter *B*) can use to send her a secret message. *A* can decrypt the secret message using her secret key, which is different from the public key. Meanwhile, an eavesdropper -- named *C* -- is trying to decode the message without having been told what the secret key is. *A* hopes that *C* will not be able to do this. That is, she hopes that the cryptosystem will withstand *C*'s attempts to find the secret message.

In our system we will suppose that the secret message *m* is an integer between 0 and 100 that is the answer to a difficult question. (For example: How old was Elvis when Lisa Marie was born? Or: How old was Simon Bolivar at the time of his victory in the Battle of Ayacucho?) Our object is for each *B* person to communicate his or her message *m* to the corresponding *A* without their ``enemy" *C* being able to decipher the message. If this happens, then *A* and *B* win; however, if *C* finds out the message, then *C* wins, and *A* and *B* lose. (*A* and *B* also lose if *B* makes an arithmetic mistake so that *A* gets the wrong message.)

Each student receives an envelope labeled *A*, *B*, or *C* that contains a slip of paper with a question and a copy of a configuration of between 15 and 25 vertices (with no edges). If the youngster is a *B*, the answer will be given along with the question; *A* and *C* are given the same question, but not the answer. The job of *B* is to convey the answer to *A* without *C* being able to figure it out.

Each envelope also contains several pages of practice examples of graphs with perfect codes. While waiting for their turn in the protocol, the students -- especially the *C*'s -- should practice looking for the perfect codes in the examples. Here is what the children in each category are told to do.

The *A*'s:

Go through the above 3-step procedure to get a graph with a secret perfect code, using the sheet of vertices that is in your envelope. When you are done, let your *B*-partner and your *C*-enemy copy your graph. Neither of them can see your secret perfect code, however.

After encoding the message, *B* will give you a copy of your graph with green numbers beside the vertices. You then copy down the green numbers that are beside the vertices of your perfect code **C**. Add up these numbers to get the message *m*.

If your graph is hard enough so that *C* is unable to find a perfect code, then the same graph can be used again. For example, have *B* and *C* interchange roles, and see if *C* can send you a message without *B* being able to break the code.

The *B*'s:

Look at the answer to your question, which will be an integer *m* between 0 and 100. Take a sheet of vertices, and write a number in blue beside each vertex. The numbers can be any integers (you may use negative numbers, but please don't use very large numbers) whose *sum* is the number *m*.

When your *A*-partner is done with his or her graph, copy it onto another sheet of vertices. Your *A*-partner will show you only the graph, not the secret perfect code. On your copy of the graph, write a number in green beside each vertex. You compute the green number at a vertex by adding together all of the blue numbers in the vertex's neighborhood. (Remember: ``neighborhood" means the vertex itself together with all neighboring vertices.) Finally, you must allow the *C*-person to copy down the same green numbers that are on your copy of the graph. *C* will try to find the secret message by looking for a perfect code in the graph.

Throw away the vertex sheet with the blue numbers, and give your *A*-partner the copy of the graph with the green numbers.

The *C*'s:

First sharpen your claws -- practice finding secret perfect codes in a graph -- by doing some of the examples in your envelope. Work on the practice examples until *A* and *B* are ready with their graph and encoded message. Copy *A*'s graph onto your sheet of vertices, and get *B*'s green numbers. Your object is to determine *m* by finding a perfect code. Of course, *B* will not tell you *m*, and *A* will not tell you the perfect code.

Try to find a perfect code in *A*'s graph. If you succeed, then you can determine the message *m* by adding up all of the green numbers that are beside the vertices of your perfect code.

Security

Finally, it should be stressed that Perfect Code Cryptography is secure only if the youngsters do not know linear algebra. If they think of letting unknowns stand for the blue numbers, then they can write equations of the form $x+y+z+...+w=g$, where g is a known green number. The unknown blue numbers can then be found by Gaussian elimination. Any student who figures this out -- that is, who rediscovers linear algebra -- can determine any secret message without having to find a perfect code.

It is a little tedious to do the linear algebra with a 15 X 15 or 25 X 25 system. But it might not be too bad, since most of the coefficients are 0 and the non-zero coefficients are all 1.

My Experience Teaching Perfect Code Cryptography

Each year during Spring break, the University of Washington Math Department invites over a thousand high school students to Math Day on the U.W. campus. This year (1997), with the help of several of my math education students and two colleagues in the Computer Science Department, I gave Perfect Code Cryptography as a Math Day activity. During three 45-minute sessions we worked with a total of over a hundred youngsters. To begin each session, we had volunteers stand on the vertices of a giant 11-vertex graph on a 20-ft X 30-ft tarp. Through trial and error, the students found the 3-vertex perfect code. Then we encrypted and decrypted a secret message through the procedure with the blue and green numbers (see above). The students then sat down, opened their envelopes, and tried out Perfect Code Cryptography in groups of three (one *A*, one *B*, and one *C* in each group). It took a lot of advance planning and careful organization to get everything done in such a short period. The workshops went well; the only complaint was that we should have had at least an hour.

Later in the Spring, I had another opportunity to try out Perfect Code Cryptography with a group of schoolchildren during a day-long visit to the Colegio Gauss (Gauss School) in Lima, Peru. An inexpensive private school in a working-class district, Colegio Gauss, as its name implies, places a special emphasis on mathematics. In some ways the circumstances were more difficult than they had been at Math Day in the U.S. I had to explain everything in Spanish, and I had only three adults to help me (rather than nine as at Math Day). But on the positive side, our time was not so limited. In fact, the youngsters continued working on the graph theory and cryptography for three hours without a break. By the end most seemed to have mastered the main points of the cryptosystem. Interestingly, in no case was *C* able to find the hidden perfect code in *A*'s graph; and even these unusually bright 12-year-olds did not think of breaking the system by solving for the blue numbers by elimination of unknowns.

References

1. M. R. Fellows, Computer science and mathematics in the elementary schools, in N. D. Fisher, H. B. Keynes, and P. D. Wagreich, editors, *Mathematicians and Education Reform 1990-1991*, Providence, RI: American Mathematical Society (1993), pp. 143-163.

2. M. R. Fellows' web pages: cs.uidaho.edu/~casey931/Mega-Math/Welcome.html and csr.uvic.ca/~mania
3. M. R. Fellows and N. Koblitz, Kid Krypto, in E. F. Brickell, editor, *Advances in Cryptology - Crypto '92*, Springer-Verlag (1993), pp. 371-389.
4. D. Kahn, *The Codebreakers*, Macmillan (1967).
5. N. Koblitz, *A Course in Number Theory and Cryptography*, 2nd ed., Springer-Verlag (1994).
6. N. Koblitz, The case against computers in K-13 math education (kindergarten through calculus), *The Mathematical Intelligencer* **18** (1996), pp. 9-16.
7. K. Rosen, *Elementary Number Theory and Its Applications*, 3rd ed., Addison-Wesley (1993).

Return to my home page.
Return to Math Department home page.