Scientific Research Publishing

Open Journal of Discrete Mathematics, 2017, 7, 13-31
http://www.scirp.org/journal/ojdm
ISSN Online: 2161-7643
ISSN Print: 2161-7635

Number in Mathematical Cryptography

Nathan Hamlin

Department of Mathematics and Statistics, Pullman, Washington, USA
Email: n.hamlin@wsu.edu

How to cite this paper: Hamlin, N. (2017) Number in Mathematical Cryptography. *Open Journal of Discrete Mathematics*, **7**, 13-31.
http://dx.doi.org/10.4236/ojdm.2017.71003

Received: November 1, 2016
Accepted: January 20, 2017
Published: January 23, 2017

Abstract

With the challenge of quantum computing ahead, an analysis of number and representation adequate to the task is needed. Some clarifications on the combinatorial nature of representation are presented here; this is related to the foundations of digital representations of integers, and is thus also of interest in clarifying what numbers are and how they are used in pure and applied mathematics. The author hopes this work will help mathematicians and computer scientists better understand the nature of the Generalized Knapsack Code, a lattice-based code which the author believes to be particularly promising, and the use of number in computing in general.

Keywords

Number Theory, Quantum Computing, Public-Key Cryptography, Generalized Knapsack Code, Combinatorial Code

1. Introduction

The Generalized Knapsack Code (GKC), first introduced in a dissertation [1], was recently shown to be secure against basis reduction attacks [2]. With basis reduction eliminated as a threat, it was shown that the kinds of combinatorial disguising methods that are then possible would make the Shamir attack [3], an admittedly formidable problem for the knapsack of Merkle and Hellman [4], very manageable. Here I clarify some of the issues surrounding number generally that perhaps are necessary to understand and appreciate this code, as well as cryptography and computing in general. My goal here is to arrive at a view of number and representation that is more or less adequate to conventional and quantum computing. The author is not attempting to challenge philosophical materialism, for which he has a great deal of respect, but he believes that something beyond the discussion of Dedekind [5] and Hilbert is necessary to grasp the role of number and representation in computing and cryptography, and that this can be accomplished without reverting to a form of Platonism. The traditional knapsack code [6] has been examined extensively. It is based on the possibility of identifying the binary representation of a number with a 0 - 1 knapsack problem. It seems that it has been assumed that the actual base of the digital representation of a

number does not play a noticeable role in contemporary cryptography. The binary representation is particularly convenient for modern computation, and so it has been assumed that the binary representation is ideal, or at least neutral. While the actual information in a contemporary communication will be most likely transmitted in binary, as we shall see below, the selection of another more complicated base does have important cryptographic implications. In particular, the base that is chosen contributes to making certain properties of the number visible or not.

Cryptography is a distinct linguistic, mathematical, and representational process from computing, as can be seen by the fact that for most of its history it was done with paper and ink, and later, the telegraph. Thus there is at the very least no reason to assume that an account of number that is sufficient for computation or electronic communication will be adequate to cryptography as whole, even when done with the assistance of computers. And because of the linguistic nature of cryptography, there is every reason to believe that a complete account of the nature of representation is required to understand it fully. Since the practical cryptography we are dealing with today is largely a matter of disguising positive integers, something less than that full account, but, as we shall see, more than the account of binary and base 10, is needed to make sense of the field as it currently stands. That is what I am attempting to give here. To put things very simplistically, if you are trying to hide an integer from others in the computing process, it might be wise to store and transmit the number in a representation that does not reveal the essential properties of that number with respect to computation. The Hamlin-Webb representations make infinitely many such representations available for our use.

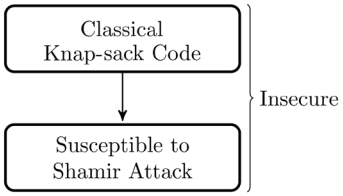
From my experience in presenting this material, I suspect this exposition will not be entirely satisfying to everyone, and especially to some who are more on the applied end of things with math and physics. I believe that, even so, some of the conclusions may be satisfactory and comprehensible to such readers. If the reader finds these arguments irritating, I suggest that he or she skip to the final paragraph of the paper.

2. A Comparison with Other Lattice-Based Codes

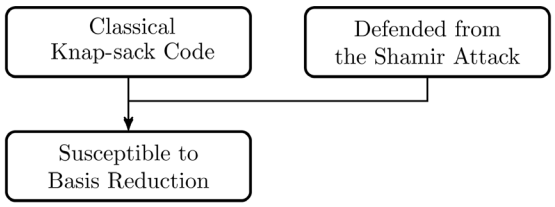
Unlike NTRU and related cryptosystems [7], the security of the GKC does not rest merely or even primarily on a closest or shortest vector problem. This makes it a fundamentally different kind of code than many other lattice-based codes, and so a different kind of explanation regarding its security is needed.

Problems with the Classical Knapsack Code

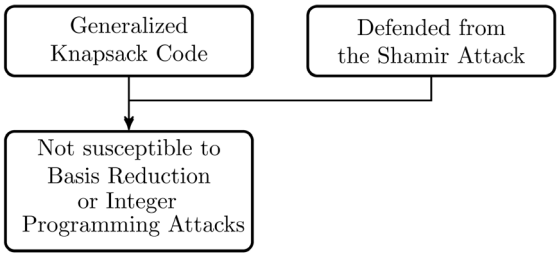
A visualization of the problems with the original knapsack code will help prepare us to understand why the GKC need not have the same vulnerabilities of other lattice based codes.



Attempts were made to defend against the Shamir attack, but the resulting change in density made another attack plausible:



The situation with the Generalized Knapsack Code is much better [2]:



Basis reduction and the Shamir attack being the primary (mathematical) security concerns, this warrants the claim that the GKC is a secure public-key code.

3. Numbers, Numerals, Images, and Representations

In order to make a compelling case for the security of the GKC within contemporary within the context of contemporary mathematical, computer science, and engineering understandings a few issues regarding representation will need to be clarified that are not always addressed in an article such as this. While some of what is said here might appear to be of use to a general audience, my goal is to clarify number for the people who use them the most, and pure mathematicians, applied mathematicians, computer scientists, and engineers all use number slightly differently; and people from each of these fields use them slightly differently. And people from all these fields are involved in mathematical cryptography. Nor, to my mind, would it be accurate to describe this a a philosophical approach to number. Philosophers have metaphysical and epistemological commitments which I am I attempting to avoid even stepping on, and I believe their cultural responsibilities go beyond questions of mere accuracy. So, I am attempting to give a relatively pragmatic approach to an issue that many people care about for different reasons, in order to show how number works in mathematical cryptography, and hopefully provide a meeting place for the different concerns and people involved. Perhaps it might be helpful to illustrate this for one important case: the general pure mathematical approach to numbers versus the applied mathematical approach. Pure mathematicians must deal with infinite, and not merely infinitesimal, quantities or even sequences and we always have, practically speaking only finitely many symbols available to us. So a pure mathematician must leave open the possibility that numbers are not only non-identical with their symbols, but perhaps independent. The applied mathematician need only take into account quantities that impact his

problems or kinds of problems, in the specific or general case, and so symbols with carefully defined meanings are of much greater importance, and these usually will ultimately refer to physical quantities, processes, and states, as opposed to entities that could be meaningfully described as abstract. By the time we get to physics, symbols can take on a life of their own, that perhaps a pure or applied mathematician might be uncomfortable with, but for the physicist extremely useful. I see no problem with any of these perspectives, but merely wish to point out that people from each area might be working together or making decisions on a problem that is primarily about number. In the case of cryptography we might be primarily interested in the natural numbers, and in my experience pure and applied mathematicians do not really agree about these. I don't wish to privilege one position over another, but to elucidate things to the point where there can some agreement here, even if that agreement cannot be elsewhere. This is important in making decisions about adopting specific public-key codes that work primarily on the natural numbers. I will include some things from the history of numbers and their representations, not as much as a survey, but rather to indicate some of the distinct sources of our perspectives on these issues. The philosophical view of number for elementary number theory which I favor is that of Husserl; but I very much believe it should not be regarded as giving the whole story. So, I am not attempting to fundamentally challenge the positions of Dedekind and Hilbert, which are very important in their own places, and which in my experience have a surprising amount of feeling attached to them in mathematical circles. Thus I will be treating this topic somewhat generally, but not for a general audience.

There is also, I believe, something very deep here, which has not generally been recognized within the mathematical and computing community, or at least been recognized as being of fundamental importance to cryptography. I will do my best to avoid being any more philosophical than necessary, but through talking with many people about this code and about numbers generally I have become convinced that an actual explanation is in fact needed here, even to talented and well-trained mathematicians. Husserl [8] argues that most human beings cannot perform basic operations with positive integers bigger than around 12 without a form of representation. This might shed some light on the dimensions of primary school multiplication tables. If you are new to thinking about this, you might think of an intelligent adult living in a community with no knowledge of mathematics, and words for numbers that do not go beyond the number 4 or 5. One could imagine that such an adult could work with sets of tools, weapons, etc. with clarity and accuracy as long as the quantities involved did not become too large. In the contemporary West, we typically teach our children how to count, and even sometimes to write numerals, before they really have need of them in normal life, so that does really work as effectively as a thought-exercise. Even if Husserl's estimate is substantially too low, which seems unlikely, or ultimately does not apply to certain people (savants, for example), a safe claim is that most of the numbers involved in the process of cryptography (and perhaps this holds true in a certain respect for computing generally) are too large for a human mind to think of and perform elementary operations upon without a system of representation (try to imagine multiplying two integers on the order of 10^{10} without some way of representing either in

symbols). It is well known of course that this can be very difficult or even impossible if you only have a poor system of numeration (e.g. Roman numerals).

The system of Roman numerals, which is often rightly ridiculed in contemporary life because of its extreme inconvenience in even basic calculations, actually had one important thing going for it, however: under Roman numerals, every natural number is given a unique name (here, a combination of letters) that does not depend essentially upon the number's mathematical properties, but only loosely on its relation to other natural numbers, and perhaps also to the idiosyncrasies of the Latin language. The base 10 system represents a number by its congruence classes modulo the powers of 10. The binary system, which does this modulo the powers of 2, is very useful for computation and communication with machines because of the easy ability to associate it with on/off switches and perhaps a high or low signal in a fiber-optic transmission. But strictly speaking, with respect to doing justice to each number, it is in many ways the worst representation possible, precisely because of this simplicity and machine-practicality. The very simplest non-constant recurrence is $u_n = 2u_{n-1}$ with $u_0 = 1$, which yields the powers of 2. Just like proving that a theorem, or finding good empirical evidence surrounding a theorem, that indicates it does or does not hold for the number 0 or 1 (or 2 for that matter) can be almost entirely irrelevant to its truth with respect to larger natural numbers or rational, algebraic, and transcendental numbers, so information about things in the very simplest case of recurrences may have no bearing at all on more complex or complicated sequences. And this simple fact seems to have been widely overlooked. The problem that the Leonardo de Pisa representation system has (and binary is just another species of this) is that it can bring you very close to making what is perhaps the fundamental intellectual error of the Middle Ages: to assume (in Aquinas's language) that *one* is convertible with *being*. It is surely no accident that the base 10 system was developed and popularized in a world already coming under the shadow of scholasticism. Saying that n is n copies of the number 1 $(n = n*1)$ does not necessarily give us any more information about n than we had before. Euclid's treatment of number theory in terms of "strips" in *the Elements*, which seems so foreign to those familiar with modern mathematics, probably contributed significantly to this confusion. Philosophically speaking, the natural numbers are individuals, as opposed to universals or images. The intellectual relationship that the greatest modern number theorists have had with them, to a thoughtful and knowledgeable mind, might be sufficient evidence of this fact, and while it is not within the realm of mathematical proof, it is within the general experience of mathematicians and students of mathematics. That *pi* more or less has an annual birthday celebration at this point is one of many things that could be adduced to this claim, which probably holds for real numbers in general, though the author will not insist on it here. Our usual way of writing the natural numbers in increasing order of cardinality is only natural in the sense that this is an extremely convenient order for our knowledge and purposes, and of course represents roughly the order in which we come to know them as children, and perhaps on a historical level to some degree as well. Cardinality is a property of number that is of at least moderate importance, of course. But the author believes it would perhaps be more entertaining than illuminating if someone were to attempt to argue

that the most important property of the numbers 1 and 2, for example, was their cardinality. The number 5 is in some ways more interesting than 4, and of course such a direct comparison is only possible up to a point. There is surely no general result than a number's worth is inversely proportional to its cardinality, and this is a much more plausible proposition than direct proportion, and would be the main way one would have try to explain the order we have on the natural numbers in terms of cardinality. The number 1 is certainly not the least important natural number.

An ordinal interpretation has a place, but not so much in cryptography. In transmitting a message M (an integer) the processes of encoding, decoding, and cryptanalysis is surely made more and not less obscure if we interpret this as the M^{th} natural number. But to choose a simple example, say the number (and not numeral) 2: even if we could list (on paper, verbally, or in a computer database) all of its mathematical properties (and of course there are infinitely many of these) this list or these properties would not be identical with the number 2, nor would they perhaps even have exhausted what might be meant by saying "2". The base 10 system labels each number by a certain specific property that that number has in relation to the number 10. This property is not the same as the number, and it is arguably not even one of its most important mathematical properties. Our English system of appellations is in a sense, more natural than the base 10 number system, because it at least gives (somewhat generally) significant information about an individual human being (usually something about his or her history). The information the base 10 representation system gives about a number is roughly how it can be used for currency, or perhaps more generally, for measurement in a metric system. The base 2 system roughly gives the information as to how a number can be used for computing. The danger, which is less likely under Roman numeration, perhaps surprisingly, is for human beings through habituation to identify the number with this property. If we were to name roses and apples after the frequency of light they reflect in full sunlight, there would be much less danger in confusing that frequency with the particular (or even type) of rose or apple, than there is in confusing the integer represented in base 10 by 209,273,356 with its base 10 representation, since roses and apples are concrete visual objects and large integers are not. How a number can act or seem to act as a universal (say in the 15 apples left on a particular tree) is probably an important philosophical and practical question, but not directly relevant to mathematical cryptography at this time, as the interpretation of the meaning of the message number M (as apples, silver coins, time left in a time-share etc.) which perhaps does not currently play a direct role in the computational process of mathematical cryptography, but rather in the human interpretation of the decoded (or potentially decoded) message.

To be precise, so as to be able understand the general case, the base 10 system identifies the number with its properties in relationship to a sequence, and while we usually say this as the powers of 10, it is more accurate to say that it is in relation to the first order recurrence sequence $u_n = 10u_{n-1}$ with initial condition $u_0 = 1$, as the base 2 is more accurately related to the sequence $u_n = 2u_{n-1}$ with $u_0 = 1$. If you are having trouble seeing this, notice that we are generally trying to find how the number makes a certain finite sequence of place values in *places* and each place one to the left of another

is 10 or 2 times that previous place, depending on the base. So, a mathematician of course will notice that this *amounts to* the powers of 10 or 2, which, perhaps sadly (in the author's opinion), are much more familiar to many mathematicians than even the simplest recurrence sequence. But the method of determining this is a recurrence relation, even though in these simple cases it yields an exponential sequence. And it is to Leonardo de Pisa that we owe both the Hindu-Arabic base 10 system (at least in its historical Western adoption) and also the Fibonacci sequence, which is enough to at least suggest that a recurrence interpretation of each may not be exactly "out of place" here.

I am going to simplify matters a bit for the sake of discussion, so let me give an example in full first so that it can be clear to the reader what I am leaving out. Let us interpret the basic information about the number that is represented in base 10 by 781 that is contained in its base 10 representation. When we write 781, in an implicitly base 10 situation, we are generally referring to the unique positive integer that is $\equiv 1 \pmod{10}$; after subtracting 1, is $\equiv 80 \pmod{100}$ (or $\equiv 8*10 \pmod{10*10}$); after subtracting another 80, is $\equiv 700 \pmod{1000}$ (or $\equiv 7*100 \pmod{10*100}$); and after subtracting another 700, is $\equiv 0$ (mod every power of 10 (or $\equiv 0*10^n \pmod{10*10^n}$, for all n). In this last case we are in particular not interested in *higher* powers of 10 than 3. It might be convenient to call this a reverse-greedy algorithm. I am going to simplify things by identifying the digits of the base 10 representation (and similarly for base 2) with the equivalence classes of the integers mod the powers of 10. While this is not precise ($781 \pmod{10^2}$ is 81 and not 8, for example), the author believes it is a helpful way to keep what is actually happening in mind, and conveniently describes the information we are generally taking from the representation.

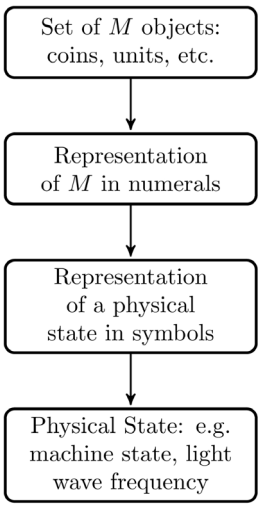
We perhaps need some new terminology. The representation classes with respect to a particular recurrence sequence are the registers in which the blocks are placed in a representation. For a first order recurrence these are just single places associated with the powers of the base. In the case of binary and base 10, these representation classes are roughly equivalence classes in each digit, and this brings a certain amount of freedom to the representation for the purposes they are used for (it is extremely convenient, for example, that adding a dime to a sum of money requires a relatively minimal change in the digits, the same perhaps could be said about shifting digits in certain conditions of the use of binary). That the representation classes for more complicated sequences are *not merely equivalence classes* is one thing that makes them superior for cryptography. The author is of course not advocating a return to classical mathematical notation. Nor does he think that our use of binary should be discontinued. But he does think it is reasonable to claim that the immense success of binary for purposes of computation *may* have made us a little overconfident with regard to its mathematical status compared to other forms of representation. (I have yet to see someone wearing a Zeckendorf t-shirt yet). Recurrence Representations are not at all likely to do what binary can do in conventional computing. But the author believes these representations can do something that binary cannot do, namely, provide (actually, infinitely many) versions of the knapsack code that are secure from basis

reduction and the Shamir attack. This is something very easy to overlook, and it appears that many people may have overlooked it.

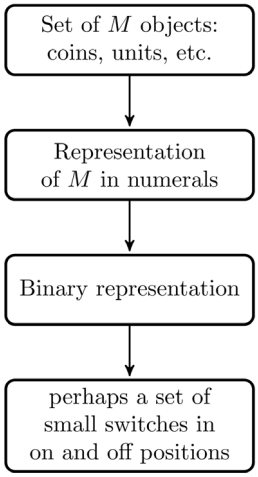
Where does this leave us with cryptography and the knapsack code? To put it bluntly, what has actually been shown in the past by other researchers is that the knapsack code, using the worst possible mathematical representation of a number (binary), is not secure from basis reduction, and thus cannot be defended against the Shamir attack. If we have followed what has been said above, this may not have any implications whatsoever for more complex or cryptographically important representations. Namely, with respect to basis reduction, representations that do not fall prey to a shortest vector approach (and any binary vector, is going to be, relatively speaking, extremely short), or a closest vector attack (and again, a binary vector with the regularities of a first-order recurrence is probably in the very easy category to approximate). So with basis reduction and integer programming, most, if not all, of the work that has been placed prominently before the scholarly community, has been done on binary representations, and I think with the assumption that this is transparent, natural, general, or at least *equivalent* to other representations. This isn't the case, and so not only are there good mathematical arguments [2] [9], and data [2] to back-up the claims that the Generalized Knapsack Code is secure from basis reduction, a simple inquiry into the nature of number and our usual representations would lead us to think that this is probably so, in light of what basis reduction and integer programming are recognized as able to accomplish. And if this is so, the arguments about providing security from the Shamir attack [1] [2] [9] with combinatorial disguising, after basis reduction is no longer a worry, are also sound and reliable. This puts us in a position of perhaps having available to us, a secure, public-key code. There seem to be good indication that this is the kind of code that is likely to also be safe under quantum computing, given at least one important inquiry into the 0 - 1 Knapsack under these conditions [10], and some analysis of the situation as a whole [6] which the author takes as suggesting, contrary to the situation with codes vulnerable to Schor's Algorithm (RSA, discrete log, elliptic curves), that the case may be that there are bounds on the possibility of finding algorithms for quantum computers that can effectively tackle some relevant lattice-based problems. To the extent which this can be currently tested, and to the extent which other concerns than security can be evaluated (for the code generally, as well as with respect to various kinds of choices of representations), I say that it is time for us to consider this code very seriously for adapting commerce and perhaps communication in light of the possibility of quantum computing. A code like this that can be implemented on conventional hardware, and yet that would be secure with a quantum computer somewhere "out there" would be much more convenient and cost effective than quantum cryptography, if the latter is to be understood as processes which require a quantum computer for one or both parties in communication.

3.1. Visualizing the Process

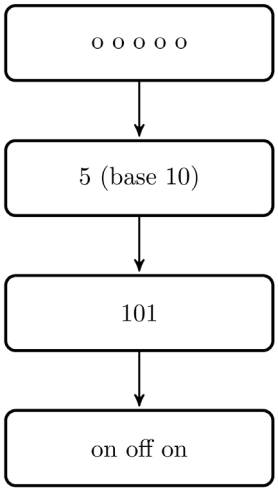
It may be helpful to see the previous and following content in terms of a diagram. We will examine the relationship between numerals, numbers, and computing with respect to both conventional and quantum computing. But first let us look at the general case:



In the case of a conventional computer running an internal operation:



In a specific case of the latter describing the cardinality of a set of 5 coins:



Of course this shows only the simplest possible cases, but it sufficiently illustrates that as we use number in computing, we have at least the numbered set (and the integer

M could be handled as M units for the purpose of visualization as in Euclid), the common human representation (for us in the West at least, base 10), the representation for the machine (in conventional computing this is often binary), the actual state of the machine that corresponds with that representation. Binary is useful because of its correspondence with the last step, but the cryptographic process involves the first two steps as well, and is at least prior to the machine state that is transmitted over the wire or wirelessly.

3.2. Numerals

I think almost everyone can agree that for human purposes, numbers generally involve a visual way of expressing them, which I will describe as a numeral. I am trying to work very carefully through some general issues in this section so that it can be clear how we use numerals while avoiding a simplistic view of numbers as forms independent of matter, which are in turn instantiated somehow in the words or characters we use for them. I think many people know that this latter outlook could have been potentially fatal for the history of mathematics at more than one point, but at the same time, within cryptography some distinction must be made between the number and the representation, if only for the fact that the representation needs to hide the number at times, and this requires at the very least that the number and representation are non-identical. So again, some general material will be presented but in terms of what is likely to be useful to a mathematician, computer scientist, and perhaps engineer. I am using the term *numeral* to refer to the visual symbol that is used to represent the number for human purposes. In the discussion which follows, which attempts to go deeper on the representation side of things, I will use a relatively small number so that some measure of visualization will be possible, but I am primarily interested in integers that are very large for my general conclusions, and this will not pose a problem to the argument. The basic thing that needs to be explored and clarified is how *numerals* function in cryptography. The author believes there are certain considerations here that were not included in say, Dedekind's and Hilbert's important discussions of these matters, and which are essential to fully understanding the GKC, and numerical representation generally. In a typical encrypted communication today, the message is a number, say a positive integer like $M=162$. Suppose this number represents something very concrete, as the number of silver coins the message sender has in a particular drawer in his or her office. The number itself is (logically at least) prior to any representation of it. If in the Western world we were still at a stage in the development of society where we could not easily represent numbers past the number 4, for example, there would still be a certain number of coins in that drawer (namely, 162; we probably need to ignore the role that numeration plays in the mint, but this has no fundamental importance here), but we might not be willing or able to record or communicate that fact, or to easily distinguish it from a situation where there were slightly more or fewer present. Even today, until someone has a great deal of familiarity with numbers, it can be difficult to meaningfully distinguish distinct large integers. Think, for example, how children will often skip from a million, to a billion, to infinite quantities so easily rather than include numbers in the sequence $\left\{10^{10},10^{100},10^{1000}\cdots\right\}$.

If coins distract from the issue at hand, think of 162 smooth stones from a river bed. If we ignore concerns related to the computer (e.g having this discussion in 1600 C.E.), the base ten representation "162" does two things for us: 1) allows us keep a good record of the coins, for ourselves or others, and 2) enables us to perform mathematical operations that will then have some important implications for the coins. Operations on "162" will have implications for the number 162 apart from the representation. Roman numerals were excellent for task 1), but not so good for task 2). To reiterate, in a certain sense, and for the purposes of cryptography I do not think we have to be very metaphysical about this, numbers exist apart from their representations. The fact that base 10 and not base 2 was chosen for this example does not change this fact, but could illustrate it: 162 and 10,100,010 refer to the same number, and apart from human convention and the certain advantages one representation might have over another in a certain context (at a sporting goods store without a cash register vs. within a bank computer records system for safety deposit boxes) they are describing the same thing. Numbers which we use are in general present in the world without representations, but we need, as thinking and relating organisms, representations in order to keep track of, communicate, and operate on, number. The fact that as mathematicians we are often dealing with infinite, unending sequences of numbers in abstract settings can make this a bit cloudy, but the finite numbers mostly at use in daily communication and cryptography typically have a much more concrete significance and are integers or at least rationals, and so the important qualifications surrounding irrationals (Dedekind) and infinite sequences and series (Hilbert, see [8] for example) are not as essential to our understanding in this context.

So without claiming that all that I say here can be applied to all mathematical settings and entities, I will continue to argue that this is true in the most important settings relevant to cryptography. The introduction of the base representational 10 system in the Middle Ages by Leonardo de Pisa, vastly improved our relationship to number, but did not change how number was present in objects (our 162 coins in the drawer for example). This new relationship did in fact give Western Europeans a certain amount of power over number and numbered things, and so perhaps there was a change in the coins, at least in the expected number to be found in the drawer at the end of the day. I only intend to point out that 162 coins is a fact found in experience that can be real before someone comes along and counts them. When you first look in the drawer, and see the coins, there is a certain number them, even before you count them. In general practice at least, the act of counting is intended at to determine a fact that is already real but currently unknown and needed to be known. I suspect that most of what I am saying here will be relatively elementary to the reader, but I am trying to be thorough enough to handle some of the philosophical objections that have been raised in our history, and which I believe have a certain point to them.

Notice that a numeral, say "1", or a collection of numerals e.g. "162" is actually an image. Its printed form, or the process of displaying it on paper or a computer screen might take the form of an algorithm involving numbers and numerals, but when we finally see it before us, we must recognize its meaning as a whole. The upper half of "1" *does not* mean 0.5 for example, and whether I draw a very small "1" in the bottom

corner of a page, or make a large poster of it, its meaning, as a number anyway, is the same. (There might be a further meaning in the former as a *page* number, and in the latter as support for a favorite, as at maybe a football playoff, and for our purposes whether the number is *understood* to be cardinal or ordinal) In cryptography at least, we are usually dealing with cardinal numbers, and this is one of the main things that makes Husserl's treatment of greater use than Hilbert's. We may break up our larger digital representation (162) into parts, but when we finally arrive at the numeral "6" we do not break it down further but take it to represent the number 6, 6 things, etc. Similarly, when faced with a photo of an apple, we do not (educational purposes perhaps excepted) divide the photo into parts, but recognize the whole as in some way representing (say) a piece of fruit we once gave to a favorite teacher or happily found in our lunch bag on a daily basis when were much younger. If it is an image we recognize, in general, "cutting it up" either mentally or with a photo program, will not increase our understanding of the image if all we need is to recognize it's meaning as a whole. Incidentally, the fact that images are in some sense *wholes* is what makes them difficult to handle for internet search engines. Probably with our current algorithms, the best that can be done is to catalogue images with labels and then search for labels. But this is not to search for images themselves; it is only like what we do in our physical libraries where we use algorithms etc. to send the appropriate location for a cover with a particular number on it. If the glue is good, we will send of course the book that belongs to that cover, but we were actually searching for the number on the cover which probably has no ultimately meaningful connection to the book itself, except in a very general manner. One cannot after all get much of a real sense of Guizot's *History of France* (a rather long work) from the information in the card catalogue (physical or automated), even if this contains a substantial summary. Thus, at least until better algorithms for images can be found, people will probably still be seen browsing at bookstores and in libraries for the foreseeable future.

Please bear with the author as I take up one possible objection here. The point I am trying to make is essential in understanding the Generalized Knapsack Code, and the theoretical underpinnings of representation of number. Images as transmitted from computer to computer or printed on paper are, with some qualifications perhaps, large, many-valued arrays with a position in 2 or 3 space matched with a color. Without having to go into finite vs. infinite kinds of discussions which are in fact appropriate here, let us be very simple, since this paper is concerned mostly with representations of numbers. The numeral "6" on a printed page or computer monitor will indeed be describable in terms of pixels and dot matrices etc. And these are essential to producing a visual image such as "6".

But the image functions as a whole in at least our understanding. Even the neural network that produces the image in our imagination as we see it on the screen and compare it to other "6"'s we've seen is not the image itself, at least as far as it functions in our understanding.

Again, I don't think we have to be particularly metaphysical here. Think of the times you have seen people from different backgrounds and cultures write their numbers differently on checks or other documents. The important thing is recognizing the "6"

however written, and if one has passed the early years of school, understanding its association with the number or a certain number of objects in a certain case.

A more rigorous treatment of this philosophically would probably be helpful for all this, but perhaps out of place in an article such as this. Here I only intend to make it clear that the numeral and the number are distinct; that the numeral functions as a whole, as an image, and is not reducible, at least in how we understand it in normal life, to an array in the sense that a machine stores it or produces it.

The author understands that the numeral "6" is probably simple enough to be handled exhaustively with algorithms etc., and will make a more a general point in this regard at a later time.

Let's apply all this to cryptography. The numeral "6" makes the number 6 present to our mind, if we have mastered our Hindu-Arabic numerals. Most of the time this is important so that we can know that there is six of something somewhere. If I am proud of my 162 silver coins in my drawer and have no fear of anyone stealing them, my message $M = 162$, or more likely, 10,100,010 (the numerals) can be sent in the clear and anyone can come to know about my collection. However, if I don't want others to know, I could send the number in a different set of numerals (perhaps resulting from a process of mathematical cryptography with algorithms) and these numerals would hide the true number of coins from anyone who intercepted the message (ideally speaking). So while for most other purposes of life, we use numerals to reveal numbers, or at least make them present, usable, or accessible, in cryptography we use them to hide, disguise, or secure. So we are using numerals for roughly speaking, the seemingly opposite purpose for which they were probably first developed. I see no problem with this of course, but I would like to give an account of how this happens so that we can understand how the representational system fits in. What is actually transmitted is a set of numerals (binary perhaps), and not a number. I understand that the coding of fiber-optic cables is designed so that these two can coincide in some way, but in at least an encrypted communication, this will not correspond to the message number. These images (numerals) either prevent someone from seeing (reading) the message, or if they have the proper knowledge to decipher (as in the case of the party I wish to communicate with), only delay, hopefully very briefly, the reading. In either case, the numerals obstruct the ordinary comprehension of unencrypted communication. So while it is indeed important to have difficult algorithms to reverse etc. the nature of representation plays a role in at least the first and last steps of all secure communication.

One could argue that it is the limitations of the strictly binary representation that made the original knapsack code insecure. It is the infinite possibilities of other representations using recurrence sequences, the basic building blocks of digital (in the most general sense of that word) representations, that makes the Generalized Knapsack Code secure. A system of representation involves more than the specific images (numerals) which represent numbers. In the base 10 system, we have numerals which represent the numbers 0 - 9 and then rules or conventions that extend the use of combinations of these numerals to represent (potentially at least) every positive integer. Thus representation is in essence a combinatorial problem, and since representation is

essential to communicating (whether openly or securely) numbers to others, the combinatorial nature of representation is essential to any form of mathematical cryptography, which is a special way, type, or action upon communication. The Generalized Knapsack Code makes use what may be the most general theory of these representations in order to hide and reveal information. The vast combinatorial possibilities that result form the basis of its security.

Let us describe the Leonardo de Pisa's representation of positive integers in full (extending this to negatives or decimals is nearly trivial). There are ten images or numerals (0, 1, 2, 3, 4, 5, 6, 7, 8, 9), as well as an understanding of what each position (their are an infinite number of these potentially, but not in any actual representation of an integer) means, namely, the nth position to the left of the decimal point holds the symbol for the number of multiples of 10^{n-1}. Thus, the base 10 representation of an integer M is an ordered collection of numerals which are understood to have a certain interpretation in terms of the powers of 10. Describing binary would be even simpler. However, base ten and binary are among the very simplest of all representations [1], and thus we do not see the full complexity that can surround representation by merely looking at them. The powers of 2 and 10 have a very simple relationship, and each digit can seem to stand on its own, and provide separate information from the rest The same would be true to some extent even of the Zeckendorf representation. So let us take a representation of full complexity and examine it a little more fully.

We will of course use the Hindu-Arabic numerals, since these are not likely to go out use anytime soon, and have the great advantage of familiarity. But it is worth noting that the manufacture or adaptation of other numerals/symbols (e.g. letters from other non-Western languages) might have implications for public-key cryptography as well.

4. An Example Representation

Let u_i satisfy the recurrence $u_n = u_{n-1} + 3u_{n-3} + 2u_{n-4}$ (the initial values are not vital, but the standard ones are generally more beautiful and convenient, *i.e.*, $u_0 = u_1 = u_2 = 1$). The signature is $S = 1032$. The representation of any positive integer M is of the form

$$M = \sum_{i=0}^{n-1} d_i u_i, \tag{1}$$

where the string of digits $d_{n-1}d_{n-2}\cdots d_1 d_0$ must be composed of blocks of digits which are lexicographically smaller than S. In this case, the allowed blocks of digits are $0,100,101,102,1030,1031$. Hence, for instance, $102|1030|0|101$ is a legitimate string, but $101|1102103$ is not. Notice that no allowed block begins with 11. The difference between the general situation and base ten or base 2 is thus two-fold: in terms of the actual position of Hindu-Arabic numerals, the number is read by finding the numeral in the nth position and multiplying it by u_{n-1}, and more importantly, not all possibilities of numerals and recurrence are used. This might seem like a truism, because the numerals we have are basically designed for base 10. But it means that these images are much more complicated. And if the recurrence is kept secret, and is one of very many possible recurrences, none of which reveal plainly useful mathematical properties of the integers, 10,210,300,101 may reveal very little if any information about

the integer it represents, or even for how it could be used for mathematical computation (addition, multiplication etc.) This latter may be difficult or even virtually impossible, but at the very least will be far from transparent.

5. Conclusions

Representations are ordered combinations of images including numerals and subscripted letters which stand for members of recurrence sequences. These latter are implicit in base 10, base 2, and others, but not in general. Thus a representation of a positive integer is an image with ordered parts, and is understood in the context of a system of representation, namely a particular recurrence sequence with its properties, and the rules that develop from its intrinsic properties. If you are keeping numbers secret, it might be smart to represent them in such a way that very little meaningful information about what you are hiding is found in that representation. Recurrence sequences are an arbitrary choice except that they are the sequences that under certain conditions retain [1] the properties that are most useful to us in the binary and base ten representations. A paper is forthcoming which introduces a similar method of cryptography to the Generalized Knapsack Code, but using more arbitrary sequences, which, if secure, might have further advantages in keeping representations obscure.

This admittedly somewhat esoteric account which I have provided will perhaps not be entirely surprising for those who understand that quantum computing is a deeper scientific problem than conventional computing, and that a deeper understanding of number and representation and their role in mathematics and computing might be required for such an advanced project. At the bare minimum, the transition to qubits takes us away from the simplicity that the binary representation provided under conventional computing (in directly corresponding to bits), and so the relevance of Recurrence Representations to modern computing may transcend the cryptographic problem. As the binary and base 10 representations are effective for conventional computing and metric measurement, so other recurrences may have their own "natural" uses. But hopefully I have provided sufficient justification to make the claim that the problem and usefulness of number and representation goes far beyond what has traditionally been understood, and that the results that have been recently advanced in this area open up the field of lattice-based cryptography significantly more widely than has been generally recognized to be likely. For those who have followed the arguments presented here and elsewhere, the only thing that really remains to be done for the Generalized Knapsack Code is some heavy testing on the scale of computer systems that are only available in a corporate or governmental context, or perhaps at some leading intellectual institutions. There is every reason to believe this will lead to a working public-key code which can be implemented on conventional or quantum computers, and which will be secure in the presence of conventional or quantum computers.

If the numbers are individuals, then their properties are not exhausted by their relationships to one another, or perhaps even by *a priori* considerations generally. And thus, we come to know each number by experience; and the numbers can be and should be the proper objects of experimentation. Computer science allows us to come to know and understand numbers that are much too big for the human mind to appreciate

directly, at least with our current modes of representation. Mathematical proof, which accomplishes a great deal in allowing us to understand and work with numbers, is not the only means to determining their properties and possibilities. And to require or be satisfied with an argument, when what is needed is actually *experiment* and *testing* is to put the cart before the horse. The arguments given above and in the citations indicate that the Generalized Knapsack Code, and perhaps the Needle-in-the-Haystack Code as well, are *very likely* to be secure public-key codes, but only through thorough experimentation with *actual numbers*, and *very large ones* at that, can we come to be reasonably satisfied that we can rely upon these codes. The test results regarding basis reduction are very favorable [2], but some large-scale implementation somewhere is needed, and one major purpose of this paper is to *appeal* for such an implementation.

Acknowledgements

Thanks to the Math Department and College of Arts and Sciences at Washington State University for making it possible for me to continue my work.

Dedication

For Ashley, who is often better about numbers, and about many other things as well. And for Thérèse, who already loves numbers, and many other good things too.

References

[1] Hamlin, N. (2016) Recurrence Representations: An Exploration of Number, Representation, and Public-Key Cryptography. Lambert Academic Publishing, Saarbrücken. (Originally published as a dissertation in 2012)

[2] Hamlin, N., Krishnamoorthy, B. and Webb, W. (2015) A Knapsack-Like Code Using Recurrence Sequence Representations. *The Fibonacci Quarterly*, **1**, 24-33.

[3] Shamir, A. (1984) A Polynomial-Time Algorithm for Breaking the Basic Merkle-Hellman Cryptosystem. *IEEE Transactions on Information Theory*, **30**, 699-704. https://doi.org/10.1109/TIT.1984.1056964

[4] Merkle, R.C. and Hellman, M.E. (1978) Hiding Information and Signatures in Trap Door Knapsacks. *IEEE Transactions on Information Theory*, **24**, 525-530. https://doi.org/10.1109/TIT.1978.1055927

[5] Dedekind, R. and Beman, W., Trans. (1909) Essays on the Theory of Numbers. The Open Court Publishing Company, Chicago.

[6] Bernstein, D., Buchmann, J. and Dahmen, E. (Eds.) (2009) Post-Quantum Cryptography. Springer, Heidelberg.

[7] Hoffstein, J., Pipher, J. and Silverman, J.H. (2008) An Introduction to Mathematical Cryptography. Springer Science, New York.

[8] Husserl, E. and Bernet, R. (Eds.), Willard, D. (Trans.) (2003) Philosophy of Arithmetic: Psychological and Logical investigations with Supplementary Texts from 1887-1901. Springer Science, Dordrecht.

[9] Hamlin, N. and Webb, W.A. (2012) Representing Positive Integers as a Sum of Linear Recurrence Sequences. *The Fibonacci Quarterly*, **50**, 99-105.

[10] Arvind, V. and Schuler, R. (2003) The Quantum Query Complexity of 0-1 Knapsack and Associated Claw Problems. *International Symposium on Algorithms and Computation*, **2906**, 168-177.

Appendix: The Generalized Knapsack Code

This is some general material which appeared in [2] and earlier in [9], and, in a different form, in [1]. Please see these sources for more extensive explanations, data, and mathematical proofs.

1) The Traditional Knapsack Code

The plaintext message is assumed to be an integer M, $0 \le M < 2^n$. We consider the representation of M in base 2:

$$M = \sum_{i=0}^{n-1} \epsilon_i 2^i, \ 0 \le \epsilon_i \le 1. \tag{2}$$

The creator of the code chooses a secret, superincreasing sequence $\{s_i\}$, *i.e.*, where $s_i > s_{i-1} + \cdots + s_0$. The secret s_i are then disguised by one or more modular multiplications of the form

$$w_i = ks_i \pmod{m}, \tag{3}$$

where k and m are kept secret. The w_i are made public. The sender of the message M computes

$$T = \sum_{i=0}^{n-1} \epsilon_i w_i. \tag{4}$$

The encoded message T is then sent over a possibly insecure channel. The hope was that only the creator of the code, who knows $k, m,$ and the s_i, can solve Equation (4) for the coefficients ϵ_i. In particular, the disguising step given in Equation (3) is easily reversed.

2) Generalized with More Generalized Representations

a) An Example Code

Let u_i satisfy the recurrence $u_{i+5} = u_{i+4} + u_{i+2} + 2u_{i+1} + 7u_i$ (the initial values are not vital, and we could take them to be the standard ones, *i.e.*, $u_0 = 1, u_1 = 1, u_2 = 1, u_3 = 2$, and $u_4 = 4$). The signature is $S = 10127$. The representation of any positive integer M is of the form

$$M = \sum_{i=0}^{n-1} d_i u_i, \tag{5}$$

where the string of digits $d_{n-1}d_{n-2}\cdots d_1 d_0$ must be composed of blocks of digits which are lexicographically smaller than S. In this case, the allowed blocks of digits are $0, 100, 1010, 1011, 10120, 10121, 10122, 10123, 10124, 10125,$ and 10126.

Hence, for instance, $1011|1023|0|1010$ is a legitimate string, but $1010|110123100$ is not. Notice that no allowed block begins with 11. We now illustrate how to calculate this type of representation of any number M using a greedy approach. Although this calculation is straightforward, the fact that makes this code harder to model for the cryptanalyst is that so many strings of digits are not allowed in the representations, even though the strings appear similar to the allowed ones, and both classes of strings use digits of the same size.

b) Computing the Representation of *M*

An easy way to calculate the representation of any number M in the recurrence sequence $\{u_i\}$ is to calculate the augmented sequence $\{v_{j,i}\}$ for $1 \le j \le 10$, $0 \le i \le n$ given by

$$\begin{aligned} v_{1,i} &= u_i, \\ v_{2,i} &= u_i + u_{i-2}, \\ v_{3,i} &= u_i + u_{i-2} + u_{i-3}, \\ v_{4,i} &= u_i + u_{i-2} + 2u_{i-3}, \\ v_{5,i} &= u_i + u_{i-2} + 2u_{i-3} + u_{i-4}, \\ &\vdots \\ v_{10,i} &= u_i + u_{i-2} + 2u_{i-3} + 6u_{i-4}. \end{aligned}$$

The $v_{j,i}$ correspond to the allowed blocks of digits. In our example, the $v_{j,i}$ occur in groups of size 10. In other examples, the groups could be much larger. The correct expression for M is found simply by using the greedy algorithm on the $v_{j,i}$, and converting the sum into an expression in the u_i. The $v_{j,i}$ could be calculated and stored, or calculated from the u_i as needed.

We explore the memory requirements for storing all the $\{v_{j,i}\}$. The principal eigenvalue of the sequence $\{u_i\}$ is $\alpha \approx 1.9754$. We may assume that u_i is roughly α^i, or is close to 2^i. After disguising, the public weights w_i will be approximately 2^{n+40}, and the target T approximately 2^{n+50}. In other words, these quantities require 40 - 50 extra bits of memory to represent. If $n = 1000$, the memory required for the weights w_i is roughly $1,040,000$ bits (or 130 kilobytes). The memory required to store all the $v_{j,i}$ is hence 1.3 megabytes. Even if n is much larger, the memory needed for the w_i is negligible.

c) Encryption and Decryption

Let $\{u_i\}$ be a recurrence sequence that satisfies the following recurrence equation.

$$u_i = a_1 u_{i-1} + a_2 u_{i-2} + \cdots + a_h u_{i-h}, \tag{6}$$

where $a_1 > 0$ and all $a_i \ge 0$. The string $\S = a_1 a_2 \cdots a_h$ is its signature, and we let $A = a_1 + a_2 + \cdots + a_h$. Every natural number N has a unique representation in the form of Equation (5), where the digits are composed of blocks that are lexicographically smaller than *S*. Including the zero block, there are A such blocks. The auxiliary sequence $\{v_{j,i}\}$ is constructed as linear combinations of the u_i with coefficients same as the blocks other than the zero block. Hence there are $A-1$ of the $v_{j,i}$ in each group. The total number of $v_{j,i}$ numbers is hence $(A-1)n$ if there are n of the u_i.

The creator of the code chooses a secret sequence $\{s_i\}$ which has the property that $s_{i+1} > s_i(u_{i+1}/u_i)$ for all i. This property replaces the condition of $\{s_i\}$ being superincreasing as used in the traditional knapsack code.

The s_i are then disguised by any invertible mapping, some of which we describe below. The resulting quantities w_i are the public weights. If M is the original plaintext message, the user of the code expresses

$$M = \sum_{i=0}^{n-1} d_i u_i = \sum_{i=0}^{n-1} \epsilon_{j,i} v_{j,i}, \tag{7}$$

and computes

$$T = \sum_{i=0}^{n-1} d_i w_i = \sum_{i=0}^{n-1} \epsilon_{j,i} y_{j,i}, \tag{8}$$

which is the transmitted message.

It is relatively easy to see that this is appropriately reversible, and [2] explores the possibilities this version has with regard to protecting against the Shamir Attack, as well as explains why the code is secure from basis reduction.