Wolfgang K. Seiler: Shor's algorithm

Sommerschule Datensicherheit, Mannheim 2003

As we have seen many times during this summer school, almost any claim about security in cryptography depends on belief: Even provable security usually only means that we can proof that the solution of some problem like breaking a code would lead to the solution of another problem which we *believe* to be hard.

Since we cannot really prove that problems like factorization or computation of discrete logarithms are hard to solve, we can only judge them from our own state of knowledge about how *we* would attack them; when choosing the parameters of a cryptosystem, we add some margin of security to take care of people who might do better then us, but we can never be sure that there has not been or will not be a break-through drastically reducing the complexity of these tasks. Hence our best choice is to select an algorithms whose breaking would solve some long standing problem in mathematics and *hope,* that anybody who makes considerable progress toward such a solution will be vain enough to publish it.

Today's talk presents yet another instance of this hope: We shall see that quantum computers can easily factor numbers and compute discrete logarithms, possibly even faster than it takes to encrypt one single block by RSA or a DL-based scheme.

As we know from Professor Schmiedmayer's talks, the physics community at universities is still very far from realizing a quantum computer, and their basic goal is not to break public key cryptography, but to gain fundamentally new insights into the laws of nature. Thus anybody achieving a real break-through toward the realization of quantum com-

puters can consider himself to be a strong candidate for a NOBEL prize in physics and/or other awards – if he publishes.

SHOR, whose algorithms shall be discussed in this talk, got the NEVANLINNA prize of the International Mathematical Union at the International Congress of Mathematicians in Berlin 1998; this prize, given for achievements of younger scientists in mathematical aspects of computer science, has been for a long time – together with the better known FIELDS-medal given to the best young mathematicians – the most prestigious award in mathematics, until two years ago, the Norwegian parliament established the ABEL-prize as an equivalent to a NOBEL-prize.

While SHOR worked on his own, any important development in the realization of quantum computers will almost certainly be achieved by a large group of scientists; the chances that all of them will keep quiet are so small that it is extremely unlikely that anybody has or will have a working quantum computer without the rest of the world's knowing. Therefore presently, SHOR's algorithms almost certainly are no threat to public key cryptography. But they very well may be in the future.

§1: Classical and quantum computations

Before we can discuss specific algorithms for quantum computers, we first have to look at the way a quantum computer works. There are several important differences to a classical computer.

a) A quantum computer is not a digital computer

In a classical computer, a bit can take the two values 0 and 1, usually encoded by the two states of a bistable switching device like a flipflop. In a quantum computer, a qbit can take infinitely many values

$$a\,|0\rangle + b\,|1\rangle\,, \qquad a, b \in \mathbb{C} \quad s.t. \quad |a|^2 + |b|^2 = 1\,.$$

These values must be input by a physical process which is subject to inaccuracies: If, for example, $|0\rangle$ and $|1\rangle$ describe a vertically and a horizontally polarized photon, the state

$$\frac{\sqrt{2}}{2}\,|0\rangle + \frac{\sqrt{2}}{2}\,|1\rangle$$

can be realized by a photon polarized with an angle of 45°; but a polarization filter can only be set to this angle up to a certain level of accuracy. Hence, the initial state of a quantum computer as well as the final measurement are usually only approximate – similar to the case of the analogue computers which were popular half a century ago. It has been shown, that an accuracy of at least 10^{-4} is sufficient to allow arbitrarily long quantum computations, and for a computation of length t an accuracy of $O(1/t)$ will do.

b) Decoherence will lead to random errors

Superposition of states is only possible as long as they do not interact with other systems, so during a quantum computation, all such interactions must be avoided. Obviously, this cannot be guaranteed absolutely, hence it is unavoidable that states will collapse or be distorted sometimes. The classical way to correct bit errors is channel coding, i.e. we store our data using an error correcting code. However, also error correcting in quantum computers is quite different from the classical case: First of all, it is impossible to duplicate an unknown quantum state; but since we start from known states, it is possible to get around this problem. Secondly, unlike a classical bit, a qbit cannot only be distorted by interchanging the values 0 and 1, but it can be moved around freely in the infinite set of its possible states, and also such errors have to be corrected. SHOR and also others have constructed quantum error correcting codes on the basis of classical codes like the HAMMING code and also shown, how computations can be done with encoded date. There is, however, a big overhead: For one logical qbit, several physical ones are needed.

c) All quantum computations are reversible

The evolution of a quantum system is described by a unitary matrix, i.e. a complex matrix U s.t. $UU^* = E$, where $U^* = {}^t\overline{U}$ is the transposed of the complex conjugate of U. In particular, any unitary matrix is invertible with $U^{-1} = U^*$, hence in quantum systems only reversible processes can occur. Most logical and arithmetical operations, however, are not invertible: The result of the multiplication $3 \cdot 4 = 12$ does not allow us

to reconstruct the factors, just as the result of

$$\mathit{true} \wedge \mathit{false} = \mathit{false}$$

could also have been caused by different inputs. Thus quantum computers cannot work with classical logic gates, of which only negation is reversible, but need other primitives which guarantee that no information is ever destroyed.

Such primitives may also be interesting if quantum computers should never exist, because such a reversible logic, according to the laws of physics, need not dissipate any energy – which may be an important aspect when computers get smaller and smaller.

The reason for this is that the entropy $S = -\sum p_i \log_2 p_i$ of information theory and the entropy $S = \int \frac{dQ}{T}$ of phenomenological thermodynamics are in fact the same physical quantity; hence decreasing the entropy requires energy, whereas its increase produces heat. An elementary calculation based on statistical physics shows, that destroying one bit of information at temperature T requires an energy of $kT \ln 2$, where $k = 1.381 \cdot 10^{-23}$ K/J is BOLTZMANN's constant. So, erasing one gigabyte of information requires about $2.4 \cdot 10^{-11}$ J of energy at room temperature, approximately the same amount one gains by eating 10^{-15} potato chips. Small as this may look, on a microscopic scale, it might be too much.

Gates for reversible or conservative logic were first studied by TOFFOLI and FREDKIN, starting around 1980. They found that gates with one or two inputs do not suffice for arbitrary computations; at least one gate with three inputs is needed. Of course, such a gate also needs at least three outputs; otherwise it would be impossible to reconstruct its input from the output.

As a first example of a gate for reversible logic let's consider the *controlled negation* CNOT defined by

$$\mathrm{CNOT}(x,y) = \begin{cases} (x, \neg y) & \text{if } x = |1\rangle \\ (x, \quad y) & \text{if } x = |0\rangle \end{cases} = (x, x \oplus y)\,,$$

where $\oplus$ denotes addition modulo two; with respect to the basis $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ it is described by the unitary matrix

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} .$$

So far it is the only quantum gate that has been experimentally realized.

An example of a gate with three inputs is TOFFOLI's gate

$$(x, y, z) \mapsto T(x, y, z) \underset{\text{def}}{=} \big((x, y, z \oplus (x \wedge y)\big) .$$

It interchanges the two vectors $|110\rangle$ and $|111\rangle$ and fixes the six other elements of the basis, hence its matrix is unitary. It can be constructed from primitives having only two inputs, which is important for its practical realization, since controlled three particle interactions are far more difficult than two particle interactions.

The special cases $T(|1\rangle, |1\rangle, z) = 1 \oplus z = \neg z$,

$$T(x, y, |0\rangle) = (x, y, x \wedge y) \quad \text{and} \quad T(\neg x, \neg y, |1\rangle) = (\neg x, \neg y, x \vee y)$$

show that, at the cost of extra input and output bits, TOFFOLI's gate can compute the classical logic functions. Since all the operations performed by a classical computer can be described in terms of *and, or* and *not,* this means that quantum computers can perform all the usual logical and arithmetical operations, but at the cost of many extra bits.

These extra bits are a problem because they cannot be simply erased. However, already in 1973, BENNETT found a way to clear them in a reversible way: Start with an input I and three sets of qbits which are cleared to zeroes. Then, retaining the input, compute the output O to one of these three areas and the unavoidable garbage bits G to another. So we have a reversible computation

$$(I, \mathbf{0}, \mathbf{0}, \mathbf{0}) \mapsto (I, O, G, \mathbf{0}) .$$

Next copy the output to the remaining area:

$$(I, O, G, \mathbf{0}) \mapsto (I, O, G, O) .$$

This again is reversible, because copying a known qbit to a qbit set to zero is basically just a controlled not. Now *undo* the reversible calculation $(I, \mathbf{0}, \mathbf{0}) \mapsto (I, O, G)$, leaving the fourth component untouched:

$$(I, O, G, O) \mapsto (I, \mathbf{0}, \mathbf{0}, O) .$$

This increases computing time by a fixed factor, but allows the garbage bits to be reused in the next step. Unfortunately, even so the number of additional bits needed can grow exponentially in the number of bits needed for a classical computation. As BENNETT and others showed later, this can be reduced by a time/memory trade-off, but it still is a problem.

§2: The algorithms

In a quantum computer, we can take two registers of length L and set each of their qbits to

$$\frac{\sqrt{2}}{2}|0\rangle + \frac{\sqrt{2}}{2}|1\rangle .$$

Then each of the two registers contains a superposition of all numbers from zero to 2^L-1. When we multiply the two registers, storing the result in a third register of length $2L$, our computer contains a superposition of all triples (a, b, c) such that $ab = c$ with $0 \le a, b \le 2^L - 1$. So, if we want to factor a product $N = pq$ with $N < 2^L$, it also contains the triple (p, q, N).

If, however, we measure the registers, we can get *any* triple (a, b, c) such that $ab = c$; we have no way to ensure that the third component will be the number N we want to factor. In fact, the probability that this happens, is less then $1/N$, which compares unfavorably to even the simplest classical factoring algorithms. In general, it is usually not possible to get at one particular component of a superposition of states.

a) Yet another modification of Fermat's method

So we must proceed in a more indirect way and measure something which will give us more information.

As we know from the talk on public key cryptography, FERMAT's approach to factorization of a given number N was to look for perfect squares among the numbers $N+x^2$ for $x=1,2,\dots$; if $N+x^2=y^2$, we have

$$N = y^2 - x^2 = (y+x)(y-x)\,.$$

The quadratic sieve and its modifications slightly generalize this method by looking for any numbers x,y such that $x^2 \equiv y^2 \bmod N$; unless $x \equiv \pm y \bmod N$, this gives factors

$$p = \gcd(x+y, N) \quad \text{and} \quad q = \gcd(x-y, N)\,.$$

SHOR uses yet another modification of this: He takes a random number x and looks for its order modulo N, that is for the smallest positive integer r such that $x^r \equiv 1 \bmod N$. If r happens to be even, we can rewrite this as

$$(x^{r/2}+1)(x^{r/2}-1) \equiv 0 \bmod N$$

and hope that

$$p = \gcd(x^{r/2}+1, N) \quad \text{and} \quad q = \gcd(x^{r/2}-1, N)$$

are proper divisors of N. If this is not the case, p and q are equal to N and 1; since r is the order of x modulo N, this means that $p=N$ and $q=1$, because $x^{r/2} \not\equiv 1 \bmod N$. Hence we are unlucky if and only if $x^{r/2} \equiv -1 \bmod N$.

For the cryptographically most interesting case that N is a product of two prime numbers, a simply but lengthy argumentation using the chinese remainder theorem shows that for a randomly chosen x this approach leads to a factorization in at least 50% of all cases; hence, by repeating it with n different numbers x, we have a success rate of at least $1-2^{-n}$, so that usually very few trials should suffice. We are left with the problem to compute the order r of a given number x modulo N.

b) Why should a quantum computer be able to find the order?

Computing the order of a number modulo a cryptographically interesting number N is close to computing a discrete logarithm modulo N and therefore infeasible on a classical computer. So, for this step, we need a quantum computer.

Let's first discuss heuristically why it should be easier for a quantum computer to compute such an order rather than to factor N directly. Consider the sequence of all numbers $x^i \bmod N$ for $i=1,2,\dots$. Then $x^{r+i}=x^i$ for all i, hence r is the period of this sequence. Using some kind of operation simulating interference, it should be possible to bring the computer into a state where we can measure this period. In fact, those familiar with FOURIER theory will know that the FOURIER transform of a sequence displays its periodicities, so we should try to perform a FOURIER-transform on a quantum state containing the superposition of many number $x^i \bmod N$.

Another hint comes from quantum theory itself: The state of a register containing L qbits is an element of a vector space V of dimension 2^L over $\mathbb{C}$ having a canonical basis consisting of the 2^L vectors $|\varepsilon_1 \dots \varepsilon_n\rangle$ with $\varepsilon_i \in \{0,1\}$. Measurements correspond to linear maps $\Lambda\colon V \to V$, and the possible results of such measurements are the eigenvalues of Λ.

If $N < 2^L$, multiplication modulo N by a fixed number x defines a linear map $V \to V$; lets look for its eigenvalues. The eigenvectors we are mostly interested in are linear combinations of the powers of x modulo N; in order to be eigenvectors, they must have coefficients reflecting the periodicity of the powers of x. So, if r is the order of x in $\mathbb{Z}/N\mathbb{Z}$, we can consider the vector

$$|\xi_k\rangle = \frac{1}{\sqrt{r}} \sum_{\nu=0}^{r-1} e^{-2\pi i \nu k/r}\, |x^m \bmod N\rangle\ ,$$

which is indeed an eigenvector with eigenvalue $e^{2\pi i k/r}$.

Measuring the eigenvalue $e^{2\pi i k/r}$ gives us the ratio k/r, and, if k and r happen to be coprime, also the order r as the denominator of k/r. The probability that two random numbers are coprime is $6/\pi^2 \approx 0.6079271$, that is approximately 60%, so there is a good chance to measure r.

If we don't measure r, i.e. if k and r have a common divisor, we get some fraction k'/r', where r' is a divisor of r; so, after repeating the experiment a couple of times, we should have a very good chance to get r as the least common multiple of the measured denominators – at least in theory.

In fact we are looking for a number r with around 600 decimal digits when we are trying to break RSA; obviously no physical experiment allows us to measure $e^{2\pi ik/r}$ with a precision which suffices to estimate r even with a precision of 600 decimal places, hence any measurement of $e^{2\pi ik/r}$ is useless.

Therefore we have to set up our quantum computation in such a way that we can measure r as the content of a quantum register, i.e. we somehow have to compute a FOURIER transform of all the states $|x^m \bmod N\rangle$.

c) The quantum Fourier transform

For a number $N < 2^L$, SHOR takes a quantum computer with two quantum registers of length $2L$ respectively L. The long one is brought into a superposition of all numbers between zero and $2^{2L}-1$, i.e. each of its qbit is set to $\frac{\sqrt{2}}{2}|0\rangle + \frac{\sqrt{2}}{2}|1\rangle$.

For the number x whose order mod N we want to compute, no quantum register is needed, since all of its bits are fixed. Therefore x is hardwired into the algorithm: We construct a series of quantum gates that will raise x to the power given by the contents of the first register modulo N in the same way as we would use FPGAs to raise x to a given power. The result is stored in the second registers. So, the quantum computer now is in the state

$$\left(|a\rangle = \frac{1}{2^L}\sum_{m=0}^{2^{2L}-1}|m\rangle\,,|y\rangle = \left|x^{|a\rangle} \bmod N\right\rangle\right)\;,$$

where $|m\rangle$ denotes the vector of binary digits of the number m.

Now we measure the second register. After the measurement it has a value $\left|x^b \bmod N\right\rangle$, where b is an ordinary integer between zero and $r-1$.

Since the whole of our computer is one quantum system, the measurement of the second register did not change the relation $x^a = y$ between the value a in the first register and the value b in the second register; since the second register now contains a fixed number $x^b \bmod N$, the contents of the first register is reduced to a superposition of those states $|m\rangle$, for

which $x^m \equiv x^b \bmod N$, i.e. those m which are congruent b modulo N. Let M be the number of such m; then the state of the first register is

$$|u\rangle = \frac{1}{\sqrt{M}}\sum_{k=0}^{M-1}|b+kr\rangle\;,$$

i.e. a superposition of a sequence of numbers with period r.

To compute r, we take the quantum FOURIER transform of the first register, i.e. we apply a linear operator replacing each number μ by

$$2^{-L}\sum_{\nu=0}^{2^{2L}-1}e^{2\pi\mu\nu/2^{2L}}\,|\nu\rangle\;.$$

This corresponds to the multiplication of the vector $|u\rangle$ by the $2^{2L}\times 2^{2L}$-matrix whose $\mu\nu$-entry is $2^{-L}e^{2\pi i\mu\nu/2^{L}}$. This matrix is unitary because of the well known orthogonality relations

$$\sum_{\nu=1}^{m}e^{2\pi i\mu\nu/m} = \begin{cases} m & \text{if } m|\mu \\ 0 & \text{otherwise}\end{cases}$$

reflecting the symmetry of m-th roots of unity around the origin. Hence performing this operation does not violate any law of quantum mechanics, and in fact an adaption of the classical fast FOURIER transform FFT shows that it can be realized by a sequence of gates each of which has only two inputs.

Applying the quantum FOURIER transform to the superimposed state $|u\rangle$ in the first register puts the latter into the state

$$\begin{aligned}&2^{-L}\sum_{\nu=0}^{2^{2L}-1}\frac{1}{\sqrt{M}}\sum_{k=0}^{M-1}e^{2\pi i(b+kr)\nu/2^{2L}}\,|\nu\rangle\\ &=\frac{1}{2^L\sqrt{M}}\sum_{\nu=0}^{2^{2L}-1}e^{2\pi ib}\left(\sum_{k=0}^{M-1}e^{2\pi irk\nu/2^{2L}}\right)|\nu\rangle\;;\end{aligned}$$

we measure this register.

Let us first discuss heuristically which result we should expect.

Since $M = \left[\frac{2^{2L}-b}{r}\right] \approx 2^{2L}/r$, the contents of the bracket is approximately equal to

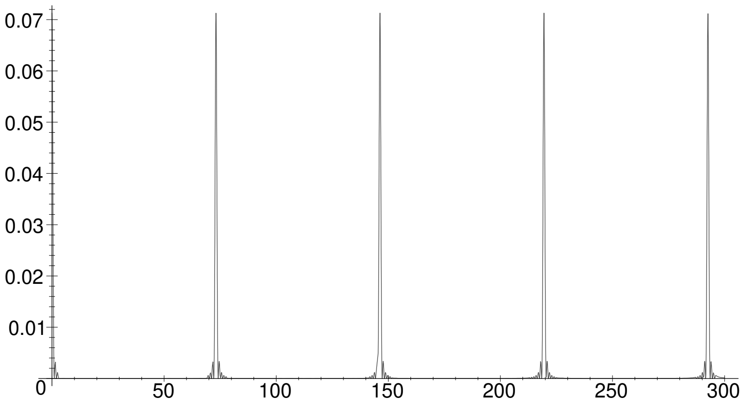
$$\sum_{k=0}^{M-1} e^{2\pi irk\nu/2^{2L}} \approx \sum_{k=0}^{M-1} e^{2\pi ik\nu/M} = \begin{cases} M & \text{if } M|\nu \\ 0 & \text{otherwise} \end{cases} .$$

Therefore we should only observe states which are approximately a multiple of $M \approx 2^{2L}/r$.

In fact, the probability to observe a number ν is, according to the laws of quantum mechanics, the square of the absolute value of its coefficient; since $e^{2\pi ib}$ has absolute value one, this probability is

$$\frac{1}{2^{2L}M}\left|\sum_{k=0}^{M-1} e^{2\pi irk\nu/2^{2L}}\right|^2 .$$

It is possible to estimate this function; here we simply look at its graph for the particular case $L = 5$ and $r = 14$, where $2^{2L}/r \approx 73.14$; we can see that there are in fact rather narrow peaks around the multiples of M.



Probability to observe a given number

Since we know L and measure (with high probability) something close to a multiple of M, we can compute something close to a divisor of r.

What we really want is of course r itself, and for this we need one more tool from mathematics.

d) Continued fractions

Continued fractions have been around for a very long time; some historians of science even argue that PLATO's academy founded the theory of real numbers on continued fractions until EUDOXOS had the better idea we still use today.

A continued fraction is a fraction written in the form

$$q = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{\cfrac{\cdots}{a_3 + \cfrac{1}{a_{n-1} + \cfrac{1}{a_n}}}}}} ;$$

it can also have infinitely many terms a_n.

Any real number x can be approximated by continued fractions using the following algorithm:

Step 1: Put $x_0 = x$ and $a_0 = [x]$.

Step n for $n \geq 1$: If $x_{n-1} = a_{n-1}$, the algorithm terminates; otherwise put

$$x_n = \frac{1}{x_{n-1} - a_{n-1}} \quad \text{and} \quad a_n = [x_n] .$$

The number q displayed above is called the n^{th} convergent of the continued fraction expansion of x.

We are interested in continued fractions because of the following

Theorem: Let $x > 0$ be a real number and suppose there are coprime integers a, b such that $\left|x - \frac{a}{b}\right| < \frac{1}{2b^2}$. Then a/b is a convergent of the continued fraction expansion of x.

As an example we can consider the continued fraction expansion of π

$$3+\cfrac{1}{7+\cfrac{1}{15+\cfrac{1}{1+\cfrac{1}{292+\cfrac{1}{\dots}}}}}$$

with convergents $\frac{22}{7}, \frac{333}{106}, \frac{355}{113}, \dots$, all of which are very good approximations of π given the size of their denominator.

Back to SHOR's algorithm! The quantum experiment gives a result μ of which is very likely to be close to a multiple of $M \approx 2^{2L}/r$. If $\mu \approx \nu M$, we have

$$M \approx \frac{\mu}{\nu} \approx \frac{2^{2L}}{r} \quad \text{or} \quad \frac{\mu}{2^{2L}} \approx \frac{\nu}{r}\,.$$

We know the fraction $\frac{\mu}{2^{2L}}$ and we know that the order r is less then 2^L; so, if $\frac{\nu}{r}$ is really close to $\frac{\mu}{2^{2L}}$, it should be a convergent of the continued fraction expansion of $\frac{\mu}{2^{2L}}$. This expansion can be easily computed and the denominators of the convergents can be checked. Since we do not get a denominator r if ν has a common divisor with r, it might be advisable to repeat the quantum experiment a couple of times and to compare the convergents given by the different results.

e) Discrete logarithms

SHOR's algorithm for computing discrete logarithms uses three quantum registers of length L, where $2^{L-1} < p < 2^L$, for the prime number p modulo which we want to compute discrete logarithms.

The first two registers are put into a uniform superposition

$$\frac{1}{\sqrt{p-1}} \sum_{m=0}^{p-2} |m\rangle$$

of all numbers from zero to $p-2$. This can be achieved by first creating a uniform superposition of all numbers up to 2^L-1, and then to measure if the register is less than $p-1$. If not, restart the computation; otherwise it is in the state we want.

Now if g is the base of the discrete logarithm and we want to compute the logarithm of x, put $g^{\text{first register}} x^{-\text{second register}} \bmod p$ into the third register; this leaves the computer in the state

$$\frac{1}{p-1} \sum_{a=0}^{p-2} \sum_{b=0}^{p-2} \left|a, b, g^a x^{-b} \bmod p\right\rangle\ .$$

Now use the quantum FOURIER transform on the first two registers to put the computer into the state

$$\frac{1}{2^L(p-1)} \sum_{\mu=0}^{2^L-1} \sum_{\nu=0}^{2^L-1} \sum_{a=0}^{p-2} \sum_{b=0}^{p-2} e^{2\pi i(\mu a+\nu b)/2^L} \left|\mu, \nu, g^a x^{-b} \bmod p\right\rangle$$

and measure the state of the computer. A rather long argument shows that sufficiently many such measurements give us a good chance to find the discrete logarithm of x; the details, however, are rather messy.

§3: Bibliography

PETER W. SHOR: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, *SIAM Rev.* **41** (1999), S. 303–332 or www.siam.org *(expanded version of the original paper)*

PETER W. SHOR: Quantum Computing, *Proceedings of the international congress of mathematicians Berlin 1998, Documenta Mathematica, Extra volume ICM* 1998 · *I,* DEUTSCHE MATHEMATIKER-VEREINIGUNG, 1998, S. 467–486 *or*
www.mathematik.uni-bielefeld.de/documenta/xvol-icm/00/00.html

A.YU. KITAEV, A.H. SHEN, M.N. VYALYI: Classical and Quantum Computation, *Graduate Studies in Mathematics* **47,** American Mathematical Society, 2002

Wolfgang K. Seiler, Mathematisches Institut, Universität Mannheim, 68131 Mannheim
seiler@uni-mannheim.de