Quantum error-correcting codes

Lectures by Harriet Pollatsek and Keldon Drudge
Notes by Peter Cameron

Draft, December 1998

Abstract

These notes are a record of proceedings in the QMW Combinatorics Study Group in November and December 1998. Since we are discrete mathematicians and know little quantum theory, the notes are not strong on the physics background (but we give references to several sources for this). We have tried to compare quantum with classical error correction where possible, and to provide enough information to see what are the workpoints for discrete mathematicians in this area.

1 Background

An exciting recent development on the border of computer science and physics has been the realisation that some important problems can be solved much more efficiently on a quantum computer than on a classical computer. Most notably, Shor [16] gave a randomised algorithm for factorising an integer in polynomial time on a quantum computer. Since factorisation of large integers is the hard problem which underpins public-key encryption systems such as RSA [14, 19], the importance of this result is obvious, and a large amount of research into the possibility of building a quantum computer is going on. (We discuss this in the Appendix.) The efficiency of a quantum computer derives from the fact that a quantum state is the superposition of a number of pure states, all of which evolve according to the laws of physics (the Schrödinger equation); in effect, a quantum computer is highly parallel.

Various schemes for a quantum computer have been proposed. The limitation derives from the fact that a bit of information is stored by a single atom or electron, instead of the large numbers of electrons used in a conventional transistor. Thus, a small nudge from the environment is sufficient to cause decoherence and upset the computation. The most promising way around this problem at present seems to be to build a fault-tolerant computer, by designing quantum error-correction into the computer to preserve the coherence of the data and the integrity of the computation. We hope to explain how error-correction could be done.

2 Bits and qubits

Classically, a single bit of information can take either the value 0 or 1, which we regard as lying in the set $\mathbb{Z}_2$. By contrast, a quantum state can be a superposition, or linear combination, of these two opposite states, with complex coefficients. Accordingly, a *qubit*, or quantum bit, lives in a 2-dimensional Hilbert space (a vector space over the complex numbers with Hermitian inner product). A state is a 1-dimensional subspace, which we normally represent by a unit vector spanning it. So we take an orthonormal basis of the space to consist of the two vectors e_0 and e_1, corresponding to the values zero and one of the qubit. An arbitrary state of the qubit is represented by $\alpha e_0 + \beta e_1$, where $|\alpha|^2 + |\beta|^2 = 1$.

According to the usual interpretation of quantum mechanics, we cannot observe the state directly. We can make a measurement, corresponding to a Hermitian (self-adjoint) operator on the space. The result of the measurement will be an eigenvalue of the operator. This result is not deterministic; different results will occur with appropriate probabilities. For example, we could measure the qubit in our example. The measurement could correspond to the operator of orthonormal projection onto the space spanned by e_1 (in matrix form, $\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$). The eigenvalues of this operator are 0 and 1, corresponding to the eigenvectors e_0 and e_1. If the system is in the state $\alpha e_0 + \beta e_1$, then we obtain the result 0 with probability $|\alpha|^2$, and the result 1 with probability $|\beta|^2$. However, the measurement changes the state of the system; after the measurement, the system is in a state spanned by an eigenvector corresponding to the eigenvalue obtained in the measurement. (If we find the value 0 for the qubit, the system will be in the state e_0, and information about α and β is lost.)

Again, in the classical case, we can represent n bits by an n-tuple of which each entry is zero or one, that is, an element of $V = \mathbb{Z}_2^n$. Correspondingly, an n-tuple of qubits lives in a complex Hilbert space having an orthonormal basis corresponding to V. We write a typical vector in V as v, and denote by e_v the corresponding basis vector of the 2^n-dimensional Hilbert space $\mathbb{C}^{2^n}$.

Note that the Hilbert space is isomorphic to the tensor product of n copies of the 2-dimensional Hilbert space in which a single qubit lives. If $v = (v_1, v_2, \ldots, v_n)$, where $v_i \in \mathbb{Z}_2 = \{0, 1\}$ for $i = 1, 2, \ldots, n$, then

$$e_v = e_{v_1} \otimes e_{v_2} \otimes \cdots \otimes e_{v_n}.$$

Why the tensor product? Peter Shor [17] says:

> One of the fundamental principles of quantum mechanics is that the joint quantum state space of two systems is the tensor product of their individual state spaces.

The *tensor product* of two spaces is the 'universal bilinear product' of the spaces. If $\{e_1, \ldots, e_m\}$ and $\{f_1, \ldots, f_n\}$ are bases for the two spaces V and W, then a basis for the tensor product $V \otimes W$ consists of all symbols $e_i \otimes f_j$, for $i = 1, \ldots, m$ and $j = 1, \ldots, n$. (Note that $\dim(V \otimes W) = \dim(V) \cdot \dim(W)$.) For $v = \sum_{i=1}^m \alpha_i e_i \in V$ and $w = \sum_{j=1}^n \beta_j f_j \in W$, we set

$$v \otimes w = \sum_{i=1}^m \sum_{j=1}^n \alpha_i \beta_j (e_i \otimes f_j).$$

Note that, in contrast to the case of a direct sum, not every vector in the tensor product can be written as a *pure tensor* $v \otimes w$.

A remark on notation. We do not use Dirac's *bra and ket notation* beloved of physicists. However, we do have to keep straight several different vector spaces carrying various forms: we already have an n-dimensional space V over $\mathbb{Z}_2$, and a 2^n-dimensional complex Hilbert space. Shortly we will meet a $2n$-dimensional $\mathbb{Z}_2$-space $\overline{E}$ with a symplectic form! We will use v, a, b for typical vectors of V; its standard basis will be $\{u_1, \ldots, u_n\}$ (where u_i is a vector with 1 in the ith position and 0 elsewhere), and the usual dot product on V will be denoted by

$$a \cdot b = \sum_{j=1}^n a_j b_j.$$

We do not give a special name to the Hilbert space. Its vectors have the form $\sum_{v \in V} \alpha_v e_v$ where $\alpha_v \in \mathbb{C}$; the inner product of $\sum \alpha_v e_v$ and $\sum \beta_v e_v$ is $\sum \overline{\alpha_v} \beta_v$.

3 Qubit errors and the error group

The evolution of a quantum system is described by a unitary transformation of the state space. We consider 'errors' to a single qubit represented by the following unitary matrices:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \qquad \textit{(bit error)}$$
$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \qquad \textit{(phase error)}$$

The effect of X is to interchange the basis vectors e_0 and e_1 (the zero and one states of the qubit). The effect of Z is to change the relative phase of the coefficients α and β of an arbitrary state (the argument of β/α) by 180°. (The arguments of α and β have no absolute significance, so we could as well use $-Z$, but the given choice is more convenient.)

Along with X and Z, we also allow the identity I (no error) and the product $Y = XZ = -ZX$. (It is more usual in quantum mechanics to set $Y = \mathrm{i}XZ$; then X, Y, Z are the standard *Pauli spin matrices*. Note that X, Z and $\mathrm{i}XZ$ are Hermitian (or self-adjoint). However, $Y = XZ$ is unitary, and then everything can be written with real coefficients. A reason to use $Y = \mathrm{i}XZ$ is that then Y is conjugate to both X and Z, which means a change of basis transforms any one to any other, and we can regard all three non-trivial errors are *equally likely*. However, we shall see that the discrete mathematics is the same whether we make this choice or the simpler real choice, so in what follows, we use $Y = XZ$.)

There is a simple expression for the effect of X and Z on the basis $\{e_0, e_1\}$. We use the convention that $(-1)^0 = 1$ and $(-1)^1 = -1$, where the exponent is taken from the finite field $\mathbb{Z}_2$. Then we have, for $v \in \mathbb{Z}_2$,

$$Xe_v = e_{v+1}, \qquad Ze_v = (-1)^{1\cdot v} e_v.$$

Now we can apply these errors 'coordinatewise' to n qubits. If u_j denotes the jth basis vector for $V = \mathbb{Z}_2^n$, then the errors to the jth qubit act on the 2^n-dimensional Hilbert space with basis $\{e_v : v \in V\}$ as

$$X(u_j) : e_v \mapsto e_{v+u_j},$$
$$Z(u_j) : e_v \mapsto (-1)^{v\cdot u_j} e_v.$$

Then we can define $X(a)$, $Z(b)$ for any vectors $a, b \in V$ by

$$X(a) : e_v \mapsto e_{v+a},$$
$$Z(b) : e_v \mapsto (-1)^{v \cdot b} e_v.$$

Then $\{X(a) : a \in V\}$ and $\{Z(b) : b \in V\}$ are groups of unitary transformations of $\mathbb{C}^{2^n}$ which are both isomorphic to the additive group of V. Together they generate the group

$$E = \langle X(a), Z(b) : a, b \in V \rangle = \{\pm X(a)Z(b) : a, b \in V\},$$

a group of order $2 \cdot 2^n \cdot 2^n = 2^{2n+1}$.

We call E the *group of allowable errors*, or, for short, the *error group*. It can also be written in the form

$$E = \{\pm A_1 \otimes \cdots \otimes A_n : A_j \in \{I, X, Y, Z\} \text{ for } j = 1, \ldots, n\}.$$

Why do we only consider these particular errors? A quantum code can correct any one-qubit error if and only if it can correct the errors $X(u_j)$, $Y(u_j)$ and $Z(u_j)$ for all j. Ths is because the matrices I, X, Y, Z span the 4-dimensional space of all 2×2 matrices: see [1, 5].

4 Forms and extraspecial 2-groups

The group E is an extraspecial 2-group, with centre $\zeta(E) = \{\pm I\}$ of order 2. We set $\overline{E} = E/\zeta(E)$, an elementary abelian group of order 2^{2n}, This is the third, and most important, vector space with which we have to deal. Before looking at it more closely, we give some details about alternating and quadratic forms on vector spaces over $\mathbb{Z}_2$ and their relation to extraspecial 2-groups.

In this section, V denotes an arbitrary vector space, not the space of n-tuples of bits introduced earlier.

Let V be a vector space over a field F. A *quadratic form* on F is a function $Q : V \to F$ which satisfies the conditions

(a) $Q(\lambda x) = \lambda^2 Q(x)$ for all $\lambda \in F$, $x \in V$.

(b) The function $B : V \times V \to F$ defined by

$$Q(x + y) = Q(x) + Q(y) + B(x, y)$$

is bilinear (that is, linear in each variable).

We express (b) by saying that the form B is obtained from Q by polarisation.

The form B defined in (b) is *symmetric*, that is, $B(x, y) = B(y, x)$. Now, if the characteristic of F is not 2, then it follows from (a) and (b) that $Q(x) = \frac{1}{2}B(x, x)$ for all $x \in V$, so that Q can be recovered from B: that is, quadratic forms and symmetric bilinear forms carry the same information. Things are very different in characteristic 2, however. We are interested in this case, specifically $F = \mathbb{Z}_2$. From now on, we assume this.

Now we find that the form B is *alternating*, that is, $B(x, x) = 0$ for all $x \in V$. In general, an alternating bilinear form is *skew-symmetric*, that is, $B(y, x) = -B(x, y)$ for all $x, y \in V$. Of course, in characteristic 2, this just says that B is symmetric.

Clearly, Q cannot be recovered from B. Instead, we see that, if Q_1 and Q_2 both polarise to B, then $Q = Q_1 - Q_2$ polarises to the zero form, that is,

$$Q(x + y) = Q(x) + Q(y).$$

Also, because $\lambda^2 = \lambda$ for all $\lambda \in F$, we have

$$Q(\lambda x) = \lambda Q(x).$$

Thus, Q is linear. Conversely, two quadratic forms differing by a linear form polarise to the same bilinear form. So each alternating bilinear form corresponds to a coset of the dual space of V in the space of all quadratic forms.

A bilinear form B is said to be *non-singular* if it has the properties

(a) if $B(x, y) = 0$ for all $y \in V$ then $x = 0$;

(b) if $B(x, y) = 0$ for all $x \in V$ then $y = 0$.

If B is skew-symmetric (or symmetric), then each of these conditions implies the other, and we need only assume one. A non-singular alternating bilinear form on V exists if and only if V has even dimension. For any such form, there is a basis $\{v_1, \ldots, v_n, w_1, \ldots, w_n\}$ for V such that

$$\begin{aligned} B(v_i, v_j) &= 0 = B(w_i, w_j) \text{ for all } i, j, \\ B(v_i, w_i) &= 1 = -B(w_i, v_i) \text{ for all } i, \\ B(v_i, w_j) &= 0 = B(w_j, v_i) \text{ for } i \neq j. \end{aligned}$$

This is called a *symplectic basis*. A linear transformation of V which preserves the form B is called *symplectic*; the *symplectic group* is the group of all such transformations.

A quadratic form on an m-dimensional vector space is *non-singular* if it cannot be written as a form in fewer than m variables by any linear change of variables. In the case of an even-dimensional vector space over $\mathbb{Z}_2$, a quadratic form Q is non-singular if and only if the bilinear form obtained by polarisation is non-singular. It can be shown that, up to symplectic transformation, there are just two types of quadratic forms associated with a given non-singular alternating bilinear form V. In terms of the symplectic basis, they are given by

$$Q_1(x) = \alpha_1\beta_1 + \cdots + \alpha_n\beta_n,$$
$$Q_2(x) = \alpha_1\beta_1 + \cdots + \alpha_n\beta_n + \alpha_n^2 + \beta_n^2,$$

where

$$x = \sum(\alpha_i u_i + \beta_i w_i).$$

Given a subspace U of V, we set

$$U^\perp = \{x \in V : B(x,u) = 0 \text{ for all } u \in U\}.$$

The non-singularity of B guarantees that

$$\dim(U) + \dim(U^\perp) = \dim V,$$

but unlike the Euclidean case it is not true in general that $V = U \oplus U^\perp$, since we may have $U \cap U^\perp \neq \{0\}$. A subspace U of V is said to be *totally isotropic* if B vanishes identically on U, in other words, if $U \leq U^\perp$.

A vector x is said to be *singular* for the quadratic form Q if $Q(x) = 0$. A subspace U is *totally singular* if Q vanishes identically on U. By polarising the restriction of Q to U, we see that a totally singular subspace is totally isotropic; but the converse is not true. (Any 1-dimensional subspace is totally isotropic, but the span of a non-singular vector is not totally singular.)

Here is a small example. Take a 2-dimensional vector space over $\mathbb{Z}_2$, with typical vector (x_1, x_2). The four quadratic forms 0, x_1^2, x_2^2 aand $x_1^2 + x_2^2 = (x_1 + x_2)^2$ are all singular, and are in fact equal to the four linear forms 0, x_1, x_2 and $x_1 + x_2$. The other four forms x_1x_2, $x_1x_2 + x_1^2 = x_1(x_1 + x_2)$, $x_1x_2 + x_2^2 = x_2(x_1 + x_2)$, and $x_1x_2 + x_1^2 + x_2^2$, are non-singular, and polarise to the bilinear form $x_1y_2 + x_2y_1$. The first three are equivalent under linear change of variable; each has value 0 at three of the four vectors and 1 at the fourth. The last form takes the value 1 at all three non-zero variables.

An *extraspecial* 2-*group* is a 2-group whose centre, derived group, and Frattini subgroup all coincide and have order 2. Such a group E has order 2^{2n+1} for some n. If $\zeta(E)$ is the centre, then we can identify $\zeta(E)$ with the additive group of $F = \mathbb{Z}_2$, and the factor group $\overline{E}$ (which is elementary abelian of order 2^{2n}) with the additive group of a $2n$-dimensional vector space over F.

Now the structure of the group can be defined in terms of the vector space. Commutation (the map $(e, f) \mapsto [e, f] = e^{-1}f^{-1}ef$) is a function from $E \times E$ to $\zeta(E)$. Observing that $[ez, f] = [e, fz] = [e, f]$ for $z \in \zeta(E)$, we see that it induces a map from $\overline{E} \times \overline{E}$ to F. It is readily checked that this map is a non-singular alternating bilinear form B. (This explains why $|\overline{E}|$ is an even power of 2.)

Moreover, squaring (the map $e \mapsto e^2$) is a function from E to $\zeta(E)$. Again, $(ez)^2 = e^2$ for $z \in \zeta(E)$, so it induces a quadratic form $Q : \overline{E} \to F$, which polarises to B.

The vector space $\overline{E}$, with the bilinear form B and quadratic form Q, determine the structure of the extraspecial group E. From the classification of quadratic forms, we conclude that there are just two extraspecial groups of order 2^{2n+1} for any n (up to isomorphism).

A subgroup S of E is normal in E if and only if it contains $\zeta(E)$. For such a subgroup, $\overline{S}$ is a subspace of $\overline{E}$. If we start with $\overline{S} < \overline{E}$ we could have the corresponding $S < E$ containing $\zeta(E)$ or not, but normality of S does not matter in what follows. The following are immediate:

(a) S is abelian if and only if $\overline{S}$ is totally isotropic;

(b) S is elementary abelian if and only if $\overline{S}$ is totally singular.

Consider our example with $n = 1$. The quadratic form x_1x_2 corresponds to a group generated by two elements of order 2 with product of order 4; this is the *dihedral group* D_8. The form $x_1x_2 + x_1^2 + x_2^2$ corresponds to a group in which all six non-central elements have order 4; this is the *quaternion group* Q_8. The singular forms correspond to groups which, while having a central subgroup of order 2 with elementary abelian quotient, are not extraspecial; x_1^2 corresponds to $C_2 \times C_4$, and 0 to $C_2 \times C_2 \times C_2$.

5 Error group, continued

In our set-up, the error group

$$E = \{\pm X(a)Z(b) : a, b \in V\}$$

is an extraspecial 2-group of order 2^{2n+1}. Its centre is $\zeta(E) = \{\pm I\}$, and $\overline{E} = E/\zeta(E)$ is a vector space over F.

We use the notation $(a|b)$ for the element of $\overline{E}$ which is the coset of $\zeta(E)$ containing $X(a)Z(b)$. (This coset is $\{X(a)Z(b), -X(a)Z(b)\}$.) We use $*$ for the bilinear form on $\overline{E}$ derived from commutation on E. In other words,

$$[X(a)Z(b), X(a')Z(b')] = (-1)^{(a|b)*(a'|b')}I.$$

A short calculation shows that

$$(a|b) * (a'|b') = a \cdot b' - a' \cdot b,$$

where $\cdot$ is the dot product on V. (Of course, we can replace the $-$ sign by a $+$ sign since the characteristic is 2.) Thus, two elements $e, f \in E$ commute if and only if $\overline{e} * \overline{f} = 0$.

The quadratic form will be denoted by Q: that is,

$$(X(a)Z(b))^2 = (-1)^{Q(a|b)}I.$$

(If we had made the choice $Y = \mathrm{i}XZ$ instead of $Y = XZ$, we would need to enlarge our group E to include $\mathrm{i}I$. As a consequence, E would have a larger center, $\zeta(E) = \{(\mathrm{i}^\ell)I : \ell = 0, 1, 2, 3\}$. Nonetheless, the factor group $E/\zeta(E)$ would be "the same" elementary abelian 2-group of order 2^{2n}, and commutation on E would give the same bilinear form on this version of $\overline{E}$. However, since it would no longer be true that $e^2 = \pm I$ for all $e \in E$, we would lose the quadratic form on $\overline{E}$.)

More on notation. The basis $\{(u_j|0), (0|u_j) : j = 1, \ldots, n\}$ is a symplectic basis for $\overline{E}$. We will reserve the symbol $\perp$ for orthogonality in this space (with respect to the form $*$). We also make another use of $\perp$, derived from this one. If S is a subgroup of E, then we let $S^\perp$ be the centraliser of S in E. (If $S' = C_E(S)$, then $\overline{S'} = \overline{S}^\perp$, so the notation is consistent.)

To avoid confusion with orthogonality in $\overline{E}$, we use a non-standard symbol for orthogonality of vectors in V. For $U \subseteq V$ we'll write $U^\ddagger = \{v \in V : v \cdot u = 0 \text{ for all } u \in U\}$.

We will use $\langle \ldots \rangle_V$, $\langle \ldots \rangle_{\overline{E}}$, $\langle \ldots \rangle_{\mathbb{C}}$ for the subspace of V, $\overline{E}$ or $\mathbb{C}^{2^n}$ respectively, spanned by a set $\ldots$; and $\langle \ldots \rangle$ (without adornment) will denote the subgroup generated by $\ldots$.

We are interested in abelian subgroups of E. Note that any set of mutually commuting transformations on the Hilbert space $\mathbb{C}^{2^n}$ which contains the adjoints of all its elements is simultaneously diagonalisable, and has an orthonormal basis consisting of eigenvectors. Said otherwise, the common eigenspaces of the transformations in the set are mutually orthogonal. Any abelian subgroup of E satisfies this.

We begin with three rather trivial examples before turning to our main illustrative example.

Example 1 $Z(V) = \{Z(b) : b \in V\}$ (the group of phase errors) is abelian; we have $\overline{Z(V)} = \{(0|b) : b \in V\}$. The common eigenspaces are the 2^n mutually orthogonal 1-dimensional spaces $\langle e_v \rangle_{\mathbb{C}}$, for $v \in V$. (This follows from the equation $Z(b)e_v = (-1)^{b \cdot v} e_v$.)

Example 2 $X(V) = \{X(a) : a \in V\}$ (the group of bit errors) is also abelian. Its eigenspaces are less trivial to describe. For any vector $b \in V$, set

$$z_b = \sum_{v \in V} (-1)^{b \cdot v} e_v.$$

Then

$$\begin{aligned} X(a)z_b &= \sum_{v \in V} (-1)^{b \cdot v} e_{v+a} \\ &= \sum_{w \in V} (-1)^{b \cdot (w+a)} e_w \\ &= (-1)^{b \cdot a} z_b, \end{aligned}$$

so z_b is a common eigenvector. A short calculation shows that these 2^n eigenvectors are mutually orthogonal; this also follows from the fact that any two of them afford different eigenvalues for at least some element of $X(V)$.

Remark As these examples suggest, if S is an abelian subgroup of E for which $\overline{S}$ is a maximal totally singular subspace of $\overline{E}$ (of dimension n), the eigenspaces for S form an orthogonal frame (a family of 2^n mutually orthogonal 1-dimensional subspaces) in $\mathbb{C}^{2^n}$. For more about the geometry of such frames, and their relation to other topics in coding theory, we refer to [2].

Example 3 Let

$$S = \{Z(b) : b \in \langle u_4, \dots, u_n \rangle_V\},$$

so

$$S^\perp = \{\pm X(a)Z(b) : b \in V, a \in \langle u_1, u_2, u_3 \rangle_V\}.$$

Then S has 2^{n-3} orthogonal eigenspaces, each of dimension $2^3 = 8$. A typical one is

$$Q_u = \langle e_{w+u} : w \in \langle u_1, u_2, u_3 \rangle_V \rangle_{\mathbb{C}},$$

where $u \in \langle u_4, \dots, u_n \rangle_V$. (This works since

$$Z(b)e_{w+u} = (-1)^{b \cdot (w+u)} e_{w+u} = (-1)^{b \cdot u} e_{w+u}$$

as $b \cdot w = 0$.) Notice that Q_u corresponds to a *character* χ_u of S, given by

$$\chi_u(Z(b)) = (-1)^{b \cdot u}.$$

Also notice that the bases of the various eigenspaces of S partition the 2^n eigenspaces of $Z(V)$ (corresponding to the maximal isotropic subspace containing $\overline{S}$). The subset of 1-spaces spanning one eigenspace Q_u consists of those affording the character χ_u: for $s \in S$, $s(e_v) = \chi_u(s)e_v$ for $v \in u + W$. (This gives another way to see that the 2^{n-3} eigenspaces of S form an orthogonal decomposition of $\mathbb{C}^{2^n}$.)

If we set $Q = Q_0$ (corresponding to the trivial or principal character), then we observe the following:

(a) For $s \in S$, $q \in Q$, we have $s(q) = q$.

(b) For $s' \in S^\perp$, $q \in Q$, we have $s'(q) \in Q$.

(c) Any $e \in E$ permutes the eigenspaces Q_u; it fixes Q if and only if it lies in $S^\perp$.

(d) There is a one-to-one correspondence between eigenspaces Q_u and characters χ_u of $\overline{S}$.

We will see that these four properties are typical.

6 Additive codes

Choose an abelian subgroup S of E. (Equivalently, choose a subspace $\overline{S}$ of $\overline{E}$ for which $\overline{S} \leq \overline{S}^{\perp}$.) Let $\dim(\overline{S}) = n - k$. Then S has 2^{n-k} eigenspaces, one for each character of $\overline{S}$. Each eigenspace has dimension 2^k, and they form an orthogonal decomposition of $\mathbb{C}^{2^n}$.

Choose Q to be the eigenspace corresponding to the trivial character. Then we have, as in the example above:

(a) For $s \neq -I$, $q \in Q$, we have $s(q) = q$.

(b) For $s' \in S^{\perp}$, $q \in Q$, we have $s'(q) \in Q$.

(c)Any $e \in E$ permutes the eigenspaces of S; it fixes Q if and only if it lies in $S^{\perp}$; so $E/S^{\perp}$ permutes the eigenspaces regularly.

(d) The eigenspaces of S are in one-to-one correspondence with the characters of $\overline{S}$: an eigenspace Q' determines a character χ of S satisfying $s(q) = \chi(s)q$ for $s \in S$ and $q \in Q'$ with $\chi(-I) = -1$ if $-I \in S$; χ thus defines a character of $\overline{S}$.

Now Q has dimension 2^k, and is isomorphic to the space of k qubits, but the embedding of Q in the larger space 'smears out' the k qubits over the space of n qubits. (This is exactly analogous to the situation in classical error-correcting codes: there, $V = \mathbb{Z}_2^n$ is the space of n bits, and $C = \{xG : x \in \mathbb{Z}_2^k\}$ is a k-dimensional linear code, which 'smears out' the k bits of x over the space V.) So Q will be our quantum error-correcting code.

In the classical case, a message consists of n bits, that is, it is a vector in V. Errors are also vectors in V: we have 'received word equals transmitted word plus error'. The zero error has no effect. If we use a code C, then errors in C are undetectable. There will be a set $\mathcal{E}$ of so-called *correctable errors*. They have the property that, if $e, f \in \mathcal{E}$ with $e \neq f$, then $e - f \notin C \setminus \{0\}$, that is, $e - f$ is not an undetectable error. This means that we can detect the addition of an error to a codeword, and we do not confuse the effects of different errors (as long as they lie in $\mathcal{E}$). The commonest situation is that when C has minimum weight at least d, when $\mathcal{E}$ consists of all words of weight at most $\lfloor (d-1)/2 \rfloor$.

Table 1 compares the situation in the classical and quantum cases.

	Classical	Quantum
Message	n bits, in $\mathbb{Z}_2^n = V$	n qubits, in $\mathbb{C}^{2^n}$
Error group	V $e(z) = z + v$	E $e(z)$ as before
Code	$C \leq V$, $\dim(C) = k$	$Q \leq \mathbb{C}^{2^n}$, $\dim(Q) = 2^k$
Undetectable errors	$C \leq V$	$S^\perp \leq E$
Errors with no effect	$\{0\} \leq C$	$S \leq S^\perp$
Correctable errors	$\mathcal{E} \subseteq V$ $e, f \in \mathcal{E} \Rightarrow$ $e - f \notin C \setminus \{0\}$	$\mathcal{E} \subseteq E$ $e, f \in \mathcal{E} \Rightarrow$ $f^{-1}e \notin S^\perp \setminus S$

Table 1: Classical and quantum error correction

Theorem 6.1 *(Calderbank* et al. *[3]) With the notation as above, assume that the minimal q-weight of* $\overline{S}^\perp \setminus \overline{S}$ *is* d*. Then* Q *corrects errors in* $\lfloor (d-1)/2 \rfloor$ *qubits.*

Here, the *quantum weight*, or q-weight, of $(a|b) \in \overline{E}$ (or, equivalently, of $\pm X(a)Z(b) \in E$) is the number of coordinates j in which either $a_j \neq 0$ or $b_j \neq 0$ (in other words, the number of qubits which have suffered a bit error, a phase error, or both.) The theorem asserts that, if $\mathcal{E}$ is the set of elements with q-weight at most $\lfloor (d-1)/2 \rfloor$, then we have $e, f \in \mathcal{E} \Rightarrow f^{-1}e \notin S^\perp \setminus S$.

Suppose Q is an additive code, i.e. Q is an eigenspace for $S \leq E$, and $\mathcal{E}$ is the set of correctable errors for Q. Here is how decoding works. Suppose the codeword $q \in Q$ is sent, but the received message is $z = e(q)$ for some $e \in \mathcal{E}$. If $s_1, \ldots, s_{n-k}$ are a set of generators for S, then by calculating $s_j(z) = \chi(s_j)z$ for $1 \leq j \leq n-k$ (the *syndrome* of e) , we can identify the character χ and therefore the eigenspace Q' containing z. We know $Q' = f(Q)$ for some $f \in \mathcal{E}$, and we decode z to $f^{-1}(z) = f^{-1}e(q)$. (Notice that this decoding step does not require knowing e or q.) In order for this procedure to lead us to the correct state $\langle q \rangle_{\mathbb{C}}$ it's necessary that $e(Q) = f(Q)$ for $e, f \in \mathcal{E}$ should imply $\langle f^{-1}e(q) \rangle_{\mathbb{C}} = \langle q \rangle_{\mathbb{C}}$. But that is exactly our condition on $\mathcal{E}$: $f^{-1}e$ cannot be in $S^\perp \setminus S$.

Now, the extra condition that distinct e and f in $\mathcal{E}$ should produce independent vectors $e(q)$ and $f(q)$ means that $f(Q) = e(Q)$ implies $f = e$. We call the code *non-degenerate* if it satisfies the stronger condition that $e, f \in \mathcal{E}$ implies $f^{-1}e \notin \overline{S}^{\perp} \setminus \{0\}$. So, in the special case of a non-degenerate code, identifying the eigenspace containing z actually identifies e (rather than just enabling e to be corrected).

Note: Calderbank *et al* [3] suggest that the dichotomy between non-degenerate and degenerate codes is less useful than one between pure and impure codes: A code is *pure* if distinct e and f in $\mathcal{E}$ give orthogonal images $e(q)$ and $f(q)$ of a codeword q. They remark that for additive codes, the two notions coincide, while in general purity implies non-degeneracy, but not conversely.

7 Main example: an $[[8,3,4]]$ code

Following the theorem in the previous section, to obtain an additive code mapping k qubits to n and correcting errors in $\lfloor (d-1)/2 \rfloor$ qubits, we need to find a self-orthogonal $(n-k)$-dimensional subspace $\overline{S}$ of the $2n$-dimensional binary space $\overline{E}$ with $\overline{S}^{\perp} \setminus \overline{S}$ having minimum q-weight d. We call such a code an $[[n,k,d]]$ code. We use the double brackets to distinguish it from a classical $[n,k,d]$ binary code, a k-dimensional subspace of an n-dimensional binary vector space whose non-zero elements have Hamming weight (H-weight) at least d. In this section we construct a specific example of an $[[8,3,4]]$ code adapted from [4] and [10].

To begin, we need a 5-dimensional subspace $\overline{S} \leq \overline{S}^{\perp}$ of the 16-dimensional binary space $\overline{E}$, and we would like the minimal non-zero q-weight of vectors in $\overline{S}^{\perp}$ to be 4.

Vectors in $\overline{E}$ have the form $(a|b)$ for $a, b \in V$, where V is a binary space of dimension 8. We base our construction of $\overline{S}$ on the $[8,4,4]$ *extended Hamming code* $\widehat{\mathcal{H}}_3$, a self-orthogonal subspace of V obtained by *extending* the $[7,4,3]$ Hamming code $\mathcal{H}_3$ by adding a parity-check bit to each vector. (Our notation reflects the fact that $[2^r-1, 2^r-1-r, 3]$ Hamming codes $\mathcal{H}_r$ exist for $r \geq 3$. There is further discussion of Hamming codes in Section 10.)

We construct $\mathcal{H}_3$ by specifying a 3×7 *parity-check matrix* H—recall the code is the null space of its parity-check matrix. The columns of H will consist of all possible non-zero vectors in $\mathbb{Z}_2^3 \simeq \mathrm{GF}(8)$. We do this by choosing a generator $\alpha = \alpha^3 + 1$ of the multiplicative group of GF(8), and letting the columns of H be $\alpha^6, \ldots, \alpha, 1$, (where $x\alpha^2 + y\alpha + z$ is written as the column $[x, y, z]^{\top}$) giving

$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

This construction of H makes it easy to see that $\mathcal{H}_3$ is a *cyclic* code: if $v = (v_1, \ldots, v_7)$ is in $\mathcal{H}_3$, so is its cyclic shift $v' = (v_7, v_1, \ldots, v_6)$. From H we easily obtain a *generator matrix* G for $\mathcal{H}_3$—recall a code is the row space of its generator matrix.

$$G_0 = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

Since the vector $\overline{1} = (1, 1, 1, 1, 1, 1, 1)$ is in $\mathcal{H}_3$, we get another generator matrix G_1 by adding $\overline{1}$ to each row of G_0.

$$G_1 = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \end{bmatrix}$$

We obtain the $[8, 3, 4]$ extended Hamming code $\widehat{\mathcal{H}}_3$ by adding a parity check bit to the front of each vector in $\mathcal{H}_3$ so that the resulting vector has even H-weight. The minimum H-weight of nonzero vectors in $\mathcal{H}_3$ is 3, and the minimum H-weight is 4 for $\widehat{\mathcal{H}}_3$. Moreover, $\widehat{\mathcal{H}}_3 = \widehat{\mathcal{H}}_3^{\ddagger}$.

Finally, we're ready to describe a 5×16 generator matrix for $\overline{S}$. Let a be the last row of G_1 and b be its first. Let a_1 be the extension of a and b_1 be the extension of b, two vectors in $\widehat{\mathcal{H}}_3$. Next a_2 is the extension of the cyclic shift a', and b_2 is the extension of b'. Similarly a_3 and b_3 are extensions of a'' and b''. We take $a_4 = b_5$ to be the 8-tuple $\overline{1}$ and $a_5 = b_4$ to be the 8-tuple $\overline{0}$. The 5×16 matrix $G^{(0)}$ has for its rows the vectors $(a_1|b_1), \ldots, (a_5|b_5)$.

$$G^{(0)} = \left[\begin{array}{cccccccc|cccccccc} 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right]$$

The rows of $G^{(0)}$ are linearly independent, so its rowspace $\overline{S}$ is of dimension 5. Because $\widehat{\mathcal{H}}_3 = \widehat{\mathcal{H}}_3^{\ddagger}$, we see that $(a_i|b_i) * (a_j|b_j) = a_i \cdot b_j + a_j \cdot b_i = 0$ for $i, j = 1, \ldots, 5$, so $\overline{S} \leq \overline{S}^{\perp}$. The dimension of $\overline{S}^{\perp}$ is $16 - 5 = 11$, and so $\overline{S}^{\perp} = \overline{S} \oplus T$ for T of dimension 6. We can choose a basis $\{(a_j|\overline{0}), (\overline{0}|b_j) : j = 1, 2, 3\}$ for T, and from this we can see that the minimum q-weight of $\overline{S}^{\perp}$ is 4.

Our code Q is the +1 eigenspace of the subgroup $S < E$ acting on the 2^8-dimensional complex vector space $\mathbf{C}^2 \otimes \ldots \otimes \mathbf{C}^2$. The subspace Q is of complex dimension 2^3; Q thus maps 3 qubits to 8 qubits and corrects errors in E that affect 1 qubit.

Finding a basis for Q. We would like a basis for Q consisting of vectors of the form $\gamma \sum_{v \in V} \alpha_v e_v$, where the coefficients $\alpha_v \in \{0, +1, -1\}$ and we make q_w a unit vector by a suitable choice of the (real) scalar γ. Every additive code Q has such a basis. The construction of the basis depends on choosing generators for $\overline{S}$ of the form $(a_1|b_1), \ldots (a_{n-k}|b_{n-k})$ with $a_1, \ldots, a_m$ linearly independent and $a_{m+1} = \ldots = a_{n-k} = \overline{0}$ (which we can always do). Then, the fact that $\overline{S}$ is totally isotropic implies that

$$\langle b_{m+1}, \ldots, b_{n-k} \rangle_V^{\ddagger} = \langle a_1, \ldots, a_m \rangle_V \oplus W$$

for some $W \leq V$ of dimension $n - (n - k - m) - m = k$. (If $n - k = m$, then $V = \langle a_1, \ldots, a_m \rangle_V \oplus W$.) If we write $s_j = X(a_j)Z(b_j) \in S$, then the desired basis for Q is $\{q_w : w \in W\}$, where

$$q_w = \frac{1}{\sqrt{2^m}} \sum_{s \in \langle s_1, \ldots, s_m \rangle} s(e_w).$$

Notice that there are $|W| = 2^k$ vectors q_w, agreeing with dim $Q = 2^k$.

First we show that the q_w are linearly independent—in fact, they are orthogonal. If we write $A = \langle a_1, \ldots, a_m \rangle_V$, then

$$q_w = \frac{1}{\sqrt{2^4}} \sum_{a \in A} \pm e_{w+a}$$

where our notation reflects the fact that the generator $s_j = X(a_j)Z(b_j)$ takes e_w to $\pm e_{w+a_j}$. Because $A \cap W = \{\overline{0}\}$, $w \neq w'$ implies $w + a \neq w' + a'$ for all $a, a' \in A$, so each q_w has for its summands a different subset of the orthonormal basis $\{e_v\}$, and q_w and $q_{w'}$ are orthogonal whenever $w \neq w'$.

Now we show that each $q_w \in Q$ by checking that $s_j(q_w) = q_w$ for each generator s_j of S. This is immediate from the definition of q_w for $j =$

$1, \ldots, m$. For $m < j \leq n-k$ notice that $w \in \langle b_{m+1}, \ldots, b_{n-k} \rangle_V^{\ddagger}$ implies $s_j(e_w) = Z(b_j)e_w = e_w$. Now use the fact that $ss_j = s_j s$ for $s \in \langle s_1, \ldots, s_m \rangle$.

In our specific case we have $m = 4$ and we may write

$$q_w = \frac{1}{\sqrt{2^4}} \sum_{s \in \langle s_1, \ldots, s_4 \rangle} s(e_w)$$

where $\langle b_5 \rangle_V^{\ddagger} = \langle a_1, \ldots, a_4 \rangle_V \oplus W$ with W of dimension 3.

8 Efficient encoding

Our encoding task is to take a pure k-qubit state $e_{(c_1, \ldots, c_k)}$ to $q_w \in Q$, where $w = \sum c_j w_j$ for some basis $\{w_1, \ldots, w_k\}$ of W, a complement of $A = \langle a_1, \ldots, a_m \rangle_V$ in $\langle b_{m+1}, \ldots, b_{n-k} \rangle_V^{\ddagger}$. We will follow Cleve and Gottesman [6] to build a quantum circuit to accomplish this encoding.

As Shor says, a *quantum circuit* is built out of "logical quantum wires," each corresponding to one of the n qubits, and *quantum gates*, each acting on one or two wires [17]. A quantum gate is a unitary transformation, since these correspond to the possible physical transformations of a quantum system. Shor assumes that each gate acts on either one or two wires; that is, each maps 1 qubit to 1 qubit or 2 qubits to 2 and acts as the identity on the remaining qubits. The Cleve and Gottesman encoding circuit is of this kind, and it is "efficient" because it uses at most $(n-k)n$ gates.

1-qubit gates We will use the 1-qubit gates X, Z and $Y = XZ$ along with another denoted H, described below. In terms of our basis e_0, e_1 for $\mathbb{C}^2$, H is defined by

$$\begin{aligned} H(e_0) &= (1/\sqrt{2})(e_0 + e_1) \quad \text{and} \\ H(e_1) &= (1/\sqrt{2})(e_0 - e_1). \end{aligned}$$

It's easy to check that H is unitary and $H^2 = I_2$.

We don't need these facts here, but notice

- $HXH = Z$;
- if $H_n = H \otimes \ldots \otimes H$ is the transformation on $\mathbb{C}^{2^n}$ obtained by applying H to each of the n qubits, then $H_n X(V) H_n = Z(V)$;

- the basis vectors $z_b = \sum_{v \in V} (-1)^{b \cdot v} e_v$ of example 2 in §5 are images of our standard basis vectors e_b under $\sqrt{2^n} H_n$; and

- the matrices with respect to our standard bases of $\sqrt{2}H$ and of $\sqrt{2^n} H_n$ are each *Hadamard* matrices, namely $r \times r$ matrices M with entries in $\{0, +1, -1\}$ satisfying $MM^t = rI$.

2-qubit gates The gates X, Y, Z and H become 2-qubit gates if they are controlled by a second qubit. The controlled version of X is particularly important, as we explain below, and it has its own name: CNOT for *controlled not*. Here is a circuit diagram for CNOT.

$$
\begin{array}{llcccc}
\text{controller} & x & — & \bullet & — & x \\
 & & & | & & \\
\text{target} & y & — & \oplus & — & x+y
\end{array}
$$

In other words, CNOT takes the basis vector $e_{(x,y)}$ of $\mathbb{C}^4$ to $e_{(x,x+y)}$. It is clearly a unitary transformation. The name *controlled not* is because the target bit y is negated (or not) according as the controller equals 1 or 0. CNOT is indeed just the controlled version of X: the CNOT circuit above takes $e_{(x,y)}$ to $(I \otimes X^x) e_{(x,y)}$, applying X or not according to the value of the controller.

We use similar representations for circuits describing our other 2-qubit gates. Here is a circuit for the controlled version of Z:

$$
\begin{array}{llccc}
\text{controller} & x & — & \bullet & — \\
 & & & | & \\
\text{target} & y & — & Z & —
\end{array}
$$

This circuit takes $e_{(x,y)}$ to $(I \otimes Z^x) e_{(x,y)}$.

It's actually possible to use CNOT as the *only* 2-qubit gate. Shor says that "CNOT together with all quantum one-bit gates forms a universal set" for any quantum circuit whose gates act on n qubits for some fixed n [17]. For a very simple example, since $Y = HXH$, the controlled version of Y using CNOT and 1-qubit gates is $(H \otimes I) \circ \mathrm{CNOT} \circ (H \otimes I)$. But working exclusively with CNOT does increase the number of 1-qubit gates required for the circuit. In what follows, we make free use of controlled versions of all of the 1-qubit gates.

The 2-qubit gates, in particular CNOT, are intrinsically global: there is no way to describe them by restricting attention to a single qubit. Because they accept as input the tensor product of *superpositions* (linear combinations) of e_0 and e_1, they are the gates that makes the computation quantum rather than classical. (Compare the discussion of Deutsch's problem in the next section.)

The following description is specific to our particular $[[8,3,4]]$ example, but the generalization to any additive $[[n,k,d]]$ code is natural.

The input to our quantum encoding circuit will be 8 qubits prepared in the state e_v with $v=(c_1,c_2,c_3,0,0,0,0,0)$. (We prepare a qubit in a given state by measuring it and obtaining the required value for the measurement.) The output will be q_w. We will accomplish this in two steps.

Step 1:

$$e_{(c_1,c_2,c_3)} \otimes e_{(0,0,0,0,0)} \to e_{(c_1,c_2,c_3)} \otimes e_0 \otimes \frac{1}{\sqrt{2^4}} \sum_{(d_1,d_2,d_3,d_4)\in\mathbb{Z}_2^4} e_{(d_1,d_2,d_3,d_4)}$$

Step 2:

$$e_{(c_1,c_2,c_3)} \otimes e_0 \otimes e_{(d_1,d_2,d_3,d_4)} \to s_1^{d_1} s_2^{d_2} s_3^{d_3} s_4^{d_4} X(w_1)^{c_1} X(w_2)^{c_2} X(w_3)^{c_3} e_{(0,0,0,0,0,0,0,0)}$$

Notice that $X(w_1)^{c_1}X(w_2)^{c_2}X(w_3)^{c_3}e_{(0,0,0,0,0,0,0,0)} = e_w$ for $w = c_1w_1 + c_2w_2 + c_3w_3$. The transformations $X(w_1), X(w_2), X(w_3)$ are called *seed generators.* They're not elements of S, but they provide the "seeds" e_w for the basis vectors q_w of Q. Equivalently, we sometimes call the vectors $(w_1|\overline{0}), (w_2|\overline{0}), (w_3|\overline{0}) \in \overline{E}$ *seed generators.*

The circuit for Step 1 is

$$
\begin{array}{cccc}
c_1 & — & — & — \\
c_2 & — & — & — \\
c_3 & — & — & — \\
0 & — & — & — \\
0 & — & H & — \\
0 & — & H & — \\
0 & — & H & — \\
0 & — & H & —
\end{array}
$$

The reason this circuit works is that

$$
\begin{aligned}
\frac{1}{\sqrt{2^4}} \sum_{d \in \mathbb{Z}_2^4} e_d &= \frac{1}{\sqrt{2^4}} (e_0 + e_1) \otimes (e_0 + e_1) \otimes (e_0 + e_1) \otimes (e_0 + e_1) \\
&= (H \otimes H \otimes H \otimes H)(e_0 \otimes e_0 \otimes e_0 \otimes e_0)
\end{aligned}
$$

Step 2 requires more work. We begin by describing how to modify the generator matrix $G^{(0)}$ for $\overline{S}$ in order first to define a basis w_1, w_2, w_3 for the complement of $\langle a_1, a_2, a_3, a_4 \rangle_V$ in $\langle b_5 \rangle_V^{\ddagger}$ and second to facillitate the construction of the circuit accomplishing Step 2.

Recall that

$$
G^{(0)} = \left[\begin{array}{cccccccc|cccccccc}
0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\
0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{array} \right]
$$

By doing row operations[1] on $G^{(0)}$, we get a new generator matrix

[1] Row(2) to (2)+(3)+(4), (3) to (3)+(4), (4) to (1)+(2)+(4), and then permute rows

$$G^{(1)} = \left[\begin{array}{cccccccc|cccccccc} 0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\ 1&1&0&1&1&0&0&0&0&1&0&1&0&0&1&1\\ 0&1&1&1&0&1&0&0&0&0&1&1&1&0&1&0\\ 1&1&1&0&0&0&1&0&0&1&0&0&1&1&1&0\\ 1&0&1&1&0&0&0&1&0&0&1&0&0&1&1&1 \end{array}\right]$$

of the form

$$\left[\begin{array}{ccc|ccc} 0 & 0 & 0 & B_1 & I_{n-k-m} & D_1 \\ A_1 & A_2 & I_m & E_1 & E_2 & F \end{array}\right]$$

where in our case $n = 8$, $n - k = 5$ is the dimension of $\overline{S}$, and $m = 4$ is the number of linearly independent a_j among the generators $(a_1|b_1)$, $\ldots$, $(a_5|b_5)$ of $\overline{S}$. The columns of $G^{(1)}$ are blocked $k, n-k-m, m$ on each side; that is A_1 and B_1 each have k columns, A_2 and E_2 each have $n-k-m$ columns, and D_1 and F each have m columns.

We can take as the "seed" generators for the basis of Q the rows of

$$\begin{aligned} G^{seed} &= \left[\begin{array}{ccc|ccc} I_k & B_1^T & 0 & 0 & 0 & 0 \end{array}\right] \\ &= \left[\begin{array}{cccccccc|cccccccc} 1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0 \end{array}\right] \end{aligned}$$

(If $n - k = m$, then we take $G^{seed} = \left[\begin{array}{cc|cc} I_k & 0 & 0 & 0 \end{array}\right]$.)

Step 2 of the encoding is described by the columns of

$$G^{aug} = \left[\begin{array}{cc} (G^{seed})^T & (G^{(1)})^T \end{array}\right] = \left[\begin{array}{c} G^{bit} \\ G^{ph} \end{array}\right]$$

In our case

$$G^{bit} = \left[\begin{array}{cccccccc} 1&0&0&0&1&0&1&1\\ 0&1&0&0&1&1&1&0\\ 0&0&1&0&0&1&1&1\\ 1&1&1&0&1&1&0&1\\ 0&0&0&0&1&0&0&0\\ 0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&1 \end{array}\right] \qquad G^{ph} = \left[\begin{array}{cccccccc} 0&0&0&1&0&0&0&0\\ 0&0&0&1&1&0&1&0\\ 0&0&0&1&0&1&0&1\\ 0&0&0&1&1&1&0&0\\ 0&0&0&1&0&1&1&0\\ 0&0&0&1&0&0&1&1\\ 0&0&0&1&1&1&1&1\\ 0&0&0&1&1&0&0&1 \end{array}\right]$$

The columns of G^{bit} are the vectors $w_1, w_2, w_3, a_5, a_1, a_2, a_3, a_4$ which determine the bit changes our circuit must accomplish. The non-zero columns of G^{ph} are b_5, b_1, b_2, b_3, b_4, which determine the phase changes it must perform.

The circuit that describes just the bit changes is

$$
\begin{array}{cccccccccccccccc}
c_1 & - & \bullet & - & - & - & - & - & \oplus & - & - & - & \oplus & - & \oplus & - \\
 & & | & & & & & & | & & & & | & & | & \\
c_2 & - & - & - & \bullet & - & - & - & \oplus & - & \oplus & - & \oplus & - & - & - \\
 & & | & & | & & & & | & & | & & | & & | & \\
c_3 & - & - & - & - & - & \bullet & - & - & - & \oplus & - & \oplus & - & \oplus & - \\
 & & | & & | & & | & & | & & | & & | & & | & \\
0 & - & \oplus & - & \oplus & - & \oplus & - & \oplus & - & \oplus & - & - & - & \oplus & - \\
 & & & & & & & & | & & | & & | & & | & \\
d_1 & - & - & - & - & - & - & - & \bullet & - & - & - & - & - & - & - \\
 & & & & & & & & & & | & & | & & | & \\
d_2 & - & - & - & - & - & - & - & - & - & \bullet & - & - & - & - & - \\
 & & & & & & & & & & & & | & & | & \\
d_3 & - & - & - & - & - & - & - & - & - & - & - & \bullet & - & - & - \\
 & & & & & & & & & & & & & & | & \\
d_4 & - & - & - & - & - & - & - & - & - & - & - & - & - & \bullet & -
\end{array}
$$

This circuit takes $e_{(c_1,c_2,c_3,0,d_1,d_2,d_3,d_4)}$ to e_v where $v = (c_1+d_1+d_3+d_4, c_2+d_1+d_2+d_3, c_3+d_2+d_3+d_4, c_1+c_2+c_3+d_1+d_2+d_4, d_1, d_2, d_3, d_4)$ as it should.

If now we put in the phase changes as well (keeping in mind that $Y = XZ$), we get

$$
\begin{array}{cccccccccccccccc}
c_1 & - & \bullet & - & - & - & - & - & \oplus & - & - & - & \oplus & - & \oplus & - \\
 & & | & & & & & & | & & & & | & & | & \\
c_2 & - & - & - & \bullet & - & - & - & Y & - & \oplus & - & Y & - & - & - \\
 & & | & & | & & & & | & & | & & | & & | & \\
c_3 & - & - & - & - & - & \bullet & - & - & - & Y & - & \oplus & - & Y & - \\
 & & | & & | & & | & & | & & | & & | & & | & \\
0 & - & \oplus & - & \oplus & - & \oplus & - & Y & - & Y & - & - & - & \oplus & - \\
 & & & & & & & & | & & | & & | & & | & \\
d_1 & - & - & - & - & - & - & - & \bullet & - & Z & - & Z & - & - & - \\
 & & & & & & & & | & & | & & | & & | & \\
d_2 & - & - & - & - & - & - & - & - & - & \bullet & - & Z & - & Z & - \\
 & & & & & & & & | & & | & & | & & | & \\
d_3 & - & Z & - & - & - & - & - & Z & - & Z & - & \bullet & - & Z & - \\
 & & & & & & & & | & & & & & & | & \\
d_4 & - & Z & - & - & - & - & - & Z & - & - & - & - & - & \bullet & -
\end{array}
$$

To see that this is correct, think of applying $s_4 = X(a_4)Z(b_4)$ last in our definition of q_w (harmless, since the s_j commute). The input to s_4, after s_1, s_2 and s_3 have had whatever effect they might (depending on d_1, d_2 and d_3), will be of the form e_r with $r = (r_1, r_2, r_3, r_4, d_1, d_2, d_3, d_4)$, where the values of the r_j could be anything, but the last four coordinates are determined. Now $Z(b_4)^{d_4} e_r = (-1)^{d_4(b_4 \cdot r)} e_r$. The dot product $b_4 \cdot r = r_3 + d_2 + d_3 + d_4$, so only the 3rd, 6th, 7th, and 8th "wires," carrying the values of r_3, d_2, d_3 and d_4 respectively, are relevant to the calculation, and they are each controlled by d_4. Isolating just this portion of the circuit,we see that the 3rd wire contributes $(-1)^{d_4 r_3}$ to the phase change. Similarly, the 6th contributes $(-1)^{d_4 d_2}$, the 7th contributes $(-1)^{d_4 d_3}$, and the 8th contributes $(-1)^{d_4 d_4}$.

For the grand finale, we put in step one of the encoding, writing $R = HZ$, to get

$$
\begin{array}{cccccccccccccccc}
c_1 & - & \bullet & - & - & - & - & - & \oplus & - & - & - & \oplus & - & \oplus & - \\
 & & | & & & & & & | & & & & | & & | & \\
c_2 & - & - & - & \bullet & - & - & - & Y & - & \oplus & - & Y & - & - & - \\
 & & | & & | & & & & | & & | & & | & & | & \\
c_3 & - & - & - & - & - & \bullet & - & - & - & Y & - & \oplus & - & Y & - \\
 & & | & & | & & | & & | & & | & & | & & | & \\
0 & - & \oplus & - & \oplus & - & \oplus & - & Y & - & Y & - & - & - & \oplus & - \\
 & & & & & & & & | & & | & & | & & | & \\
d_1 & - & H & - & - & - & - & - & \bullet & - & Z & - & Z & - & - & - \\
 & & & & & & & & | & & | & & | & & | & \\
d_2 & - & H & - & - & - & - & - & - & - & \bullet & - & Z & - & Z & - \\
 & & & & & & & & | & & | & & | & & | & \\
d_3 & - & R & - & - & - & - & - & Z & - & Z & - & \bullet & - & Z & - \\
 & & & & & & & & | & & & & & & | & \\
d_4 & - & R & - & - & - & - & - & Z & - & - & - & - & - & \bullet & -
\end{array}
$$

In our specific example, the number of 1-qubit gates is 4 and the number of 2-qubit gates is 23. In general, the number of 1-qubit gates is $m \leq n - k$ and the number of 2-qubit gates is bounded by $(n-k-m)k + (n-1)m \leq (n-1)(n-k)$, so the total number of gates is bounded by $n(n-k)$. The bound on the number of 2-qubit gates is obtained by observing that there are at most $(n-k-m)$ targets controlled by each of $c_1, \ldots, c_k$, contributing at most $(n-k-m)k$ gates. And there are at most $(n-1)$ targets controlled by each of $d_1, \ldots, d_m$, contributing at most $(n-1)m$ gates.

9 Deutsch's problem

Imagine having a "quantum black box," a function

$$f : \{0,1\} \to \{0,1\}.$$

We would like to know not the local values $f(0)$ and $f(1)$ but whether or not f has the *global* property of being constant. Can we decide this without running the black box twice? This is Deutsch's problem.

If we use a classical computer, the answer is no. But if we can use a quantum computer, the answer is yes, using Deutsch's algorithm.

The heart of Deutsch's algorithm is a 2-qubit gate reminiscent of CNOT. We define a function

$$U_f : \mathbb{C}^2 \otimes \mathbb{C}^2 \to \mathbb{C}^2 \otimes \mathbb{C}^2$$

on basis vectors by $U_f(e_x \otimes e_y) = e_x \otimes e_{y+f(x)}$. It's clear that U_f is a unitary transformation. As noted earlier, because this is a quantum computer, the input to U_f can be a tensor product of superpositions of e_0 and e_1.

Deutsch's algorithm to solve his problem is described by the following circuit.

$$\begin{array}{ccccccccc} e_0 & — & H & — & \bullet & — & H & — & \text{measure} \\ & & & & | & & & & \\ e_1 & — & H & — & U_f & — & & — & — \end{array}$$

This circuit calculates

$$[P_0 \circ (H \otimes I) \circ U_f \circ (H \otimes H)](e_0 \otimes e_1),$$

where $P_0(e_x \otimes e_y) = e_x$.

The result of the circuit before measuring is

$$\begin{array}{ll} \pm e_0 \otimes (1/\sqrt{2})(e_0 - e_1) & \text{if } f \text{ is constant, and} \\ \pm e_1 \otimes (1/\sqrt{2})(e_0 - e_1) & \text{otherwise} \end{array}$$

So, after measuring we find that we obtain the result $\pm e_0$ if f is constant, and $\pm e_1$ otherwise. Note that running the circuit once is all that is required, which means calculating f only once suffices.

This small example captures the idea behind quantum parallelism: applying U_f once to the tensor product of superpositions $(e_0 + e_1) \otimes (e_0 - e_1)$ generates a state

$$[(-1)^{f(0)} e_0 + (-1)^{f(1)} e_1] \otimes (e_0 - e_1)$$

that encodes global propeties of f.

10 Hamming codes

We are going to construct an infinite family of codes which correct one qubit error. These codes, which have the parameters $[[n, n-2m, 3]]$, where $n = (4^m - 1)/3$, are the quantum analogue of Hamming codes. We begin with a short account of classical Hamming codes (which, incidentally, were first constructed as factorial designs in statistics by Fisher [8, 9]).

Let q be a prime power, and m a positive integer. The vector space $\mathrm{GF}(q)^m$ contains $q^m - 1$ non-zero vectors, which fall into $n = (q^m - 1)/(q-1)$ one-dimensional subspaces. These subspaces are the points of the *projective space* $\mathrm{PG}(m-1, q)$. Equivalently, a set of non-zero vectors which is maximal with respect to the property that none of its elements is a scalar multiple of another has size n, containing one vector from each one-dimensional subspace. Let H be the $m \times n$ matrix whose columns are the n vectors in such a set, and let C be the code with *parity check matrix* H: that is,

$$C = \{v \in \mathrm{GF}(q)^n : vH^\top = 0\}.$$

Note that different choice of the n vectors, or writing the vectors in a different order, gives an equivalent code. (Two codes are said to be *equivalent* if one is obtained from the other by permuting the coordinates and multiplying the entries in each coordinate by a non-zero scalar.) The code C is the *Hamming code* of length n and dimension $n - m$ over $\mathrm{GF}(q)$, written $\mathcal{H}_m(q)$. Since any two columns of H are linearly independent, we see that the Hamming code has minimum weight 3.

The code we actually require, however, is the *dual Hamming code* $\mathcal{D}_m(q)$, the code which has H as its *generator matrix*: this is the row space of H, given by

$$\mathcal{D}_m(q) = \{xH : x \in \mathrm{GF}(q)^m\}.$$

On the vector space $\mathrm{GF}(q)^n$, we have the standard inner product, given by

$$v \cdot w = \sum_{j=1}^{n} v_j w_j.$$

In addition, if the field order q is a square, say $q = r^2$, then there is a field automorphism $\alpha \mapsto \overline{\alpha} = \alpha^r$ of order 2. In this case, there is also a *Hermitian inner product* on $\mathrm{GF}(q)^n$, given by

$$v \circ w = \sum_{j=1}^{n} \overline{v_j} w_j.$$

A code C is *self-orthogonal* (with respect to an inner product) if the inner product vanishes identically on C, that is, if $C \le C^\perp$ (with respect to the inner product).

Theorem 10.1 *If $q = 4$, $m \geq 2$, then $\mathcal{D}_m(q)$ is self-orthogonal with respect to the Hermitian inner product on $\mathrm{GF}(q)^n$.*

Proof The field GF(4) can be written as $\{0, 1, \omega, \overline{\omega}\}$ where $\overline{\omega} = \omega^2 = 1 + \omega$. Note that the cube of every non-zero element is equal to 1, since the multiplicative group has order 3.

The columns of the matrix H form a set S of n vectors, one in each one-dimensional subspace of $V = \mathrm{GF}(4)^m$. Now any codeword of $C = \mathcal{D}_m(q)$ is obtained by taking a linear form f on $V = \mathrm{GF}(4)^m$ and writing down its values on the elements of S. (For the rows of H are obtained in this way from the forms $f_1, \ldots, f_m$, where $f_i(\alpha_1, \ldots, \alpha_m) = \alpha_i$. These forms are a basis for the dual space V' of V, so any element of V' is a linear combination of them, and the procedure produces the corresponding linear combination of the rows.)

Now let the codeword v be obtained from the form f. Then

$$v \circ v = \sum_{s \in S} \overline{f(s)} f(s) = \sum_{s \in S} f(s)^3.$$

Now $f(s) = 0$ if and only if s lies in a $(m-1)$-dimensional subspace of $\mathrm{GF}(4)^m$ (the kernel of f). For the $(4^m - 1)/3 - (4^{m-1} - 1)/3 = 4^{m-1}$ vectors s not in $\ker(f)$, we have $f(s)^3 = 1$. Since there are an even number of such vectors, and the characteristic is 2, we have $v \circ v = 0$ for all $v \in C$.

Now take any $v, w \in C$. Then, for any $\alpha \in \mathrm{GF}(4)$, we have

$$0 = (v + \alpha w) \circ (v + \alpha w) = (\alpha + \overline{\alpha}) v \circ w.$$

Choosing $\alpha = \omega$, we have $\alpha + \overline{\alpha} \neq 0$, so $v \circ w = 0$. Thus C is self-orthogonal.

Example Take $m = 2$. Then a generator matrix for $\mathcal{D}_2(4)$ is

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & \omega & \overline{\omega} \end{bmatrix},$$

and the self-orthogonality is easily checked.

Remark A similar proof (left as an exercise) shows that the codes $\mathcal{D}_m(2)$ (for $m \geq 3$) and $\mathcal{D}_m(3)$ (for $m \geq 2$) are self-orthogonal with respect to the standard inner product on $\mathrm{GF}(q)^n$, where $q = 2, 3$.

The argument fails for other field orders. Indeed, we have only defined the Hamming codes up to equivalence; and multiplying a coordinate by an arbitrary non-zero scalar preserves self-orthogonality only in the three cases $q = 2, 3$ (with the standard inner product) or $q = 4$ (with the Hermitian inner product). For multiplication of v_i by α has the effect of multiplying the contribution of the ith coordinate to the inner product by α^2 (for the usual inner product) or α^{r+1} (for the Hermitian inner product, where $q = r^2$). Now the only fields where this factor is 1 for all choices of α are those given.

It can be shown, using the theory of cyclic codes, that a suitable choice of 'dual Hamming code' can be made from the equivalence class of such codes so that it is self-orthogonal (with respect to the appropriate inner product) in all cases except $q = 2$, $m = 2$. We refer the reader to MacWilliams and Sloane [12] for details.

The Hamming code $\mathcal{H}_m(q)$ has minimum weight 3, since it is 1-error-correcting (any two columns of its parity check matrix are linearly independent). The dual code $\mathcal{D}_m(q)$ has minimum weight q^{m-1}, since its non-zero words are defined by non-zero elements of V', each of which vanishes on $(q^m - 1)/(q - 1) - (q^{m-1} - 1)/(q - 1) = q^{m-1}$ one-dimensional subspaces of V.

11 Quantum Hamming codes

Recall that we are trying to construct a totally isotropic subspace $\overline{S}$ of $\overline{E}$ (a $2n$-dimensional vector space over $\mathbb{Z}_2 = \mathrm{GF}(2)$) such that $\overline{S}^{\perp} \setminus \overline{S}$ has large q-weight, and $\dim(\overline{S}) = n - k$.

First we construct yet another vector space. This is an n-dimensional vector space over the field $F = \mathrm{GF}(4)$. Since ω and $\overline{\omega}$ form a basis for F as vector space over GF(2), we can write any vector in this space in the form $\omega a + \overline{\omega} b$, where a and b are vectors of length n over GF(2). In other words, $a, b \in V$. Now the map $\phi : \overline{E} \to F^n$ defined by

$$\phi((a|b)) = \omega a + \overline{\omega} b$$

is a bijection, and is an isomorphism of GF(2)-vector spaces (if we regard F^n as a GF(2)-space by restricting scalars). Since $\omega a_j + \overline{\omega} b_j \neq 0$ if and only if either $a_j \neq 0$ or $b_j \neq 0$, we have a very important property of ϕ: the q-weight of a vector $(a|b) \in \overline{E}$ is equal to the Hamming weight of its image $\phi((a|b)) = \omega a + \overline{\omega} b \in F^n$.

As in the last section, let $\circ$ be the Hermitian inner product on F^n. There is a *trace map* Tr from F to GF(2) defined by

$$\mathrm{Tr}(\alpha) = \alpha + \overline{\alpha},$$

so that $\mathrm{Tr}(0) = \mathrm{Tr}(1) = 0$ and $\mathrm{Tr}(\omega) = \mathrm{Tr}(\overline{\omega}) = 1$. The trace map is linear over GF(2). Now the map $(x, y) \mapsto \mathrm{Tr}(x \circ y)$ takes pairs of vectors in F^n to GF(2). We have:

$$\mathrm{Tr}((\omega a + \overline{\omega} b) \circ (\omega a' + \overline{\omega} b')) = a \cdot b' + a' \cdot b = (a|b) * (a'|b').$$

The proof of this fact is just calculation. Using the fact that the Hermitian inner product is linear in the first variable and semilinear in the second, and the fact that $a \circ b = a \cdot b$ for $a, b \in \mathrm{GF}(2)^n$, we have

$$(\omega a + \overline{\omega} b) \circ (\omega a' + \overline{\omega} b') = a \cdot a' + \overline{\omega} a \cdot b' + \omega b \cdot a' + b \cdot b'.$$

Taking the trace now gives the result (using the linearity of trace and the fact that $\mathrm{Tr}(1) = 0$ and $\mathrm{Tr}(\omega) = \mathrm{Tr}(\overline{\omega}) = 1$).

Next we show that a subspace of F^n is totally isotropic (with respect to $\circ$) if and only if the corresponding subspace of $\overline{E}$ is totally isotropic (with respect to $*$). The forward implication is clear. So suppose that $W \leq F^n$ is the image under ϕ of a totally isotropic subspace of $\overline{E}$. This means that $\mathrm{Tr}(x \circ y) = 0$ for all $x, y \in W$. Take $x, y \in W$, and let $x \circ y = \alpha$. Then $\mathrm{Tr}(\alpha) = 0$, and

$$\mathrm{Tr}(\omega\alpha) = \mathrm{Tr}((\omega x) \circ y) = 0$$

(since $\omega x \in W$); it follows that $\alpha = 0$.

To summarise: the space F^n, on restriction of scalars, becomes isomorphic to $\overline{E}$, by a map ϕ taking q-weight to Hamming weight; and a subspace of F^n is the image of a totally isotropic subspace of $\overline{E}$ (with respect to $*$) if and only if it is totally isotropic (with respect to $\circ$). Of course, not every totally isotropic subspace of $\overline{E}$ corresponds to a subspace of W. (The image under ϕ of a subspace of $\overline{E}$ is a GF(2)-subspace but not necessarily a GF(4)-subspace.)

So, in order to construct a quantum error-correcting code, we need a subspace C of F^n such that $C^\perp \setminus C$ has large minimum Hamming-weight (where $C^\perp$ is defined by the Hermitian inner product $\circ$).

From the last section, we know that $C = \mathcal{D}_m(4)$ fills the bill. It has length $n = (4^m - 1)/3$, dimension m over GF(4) (hence $2m$ over GF(2)), satisfies $C \leq C^\perp$, and [assuming something suitable inserted in the previous section]

$C^\perp \setminus C$ has minimum H-weight 3. So we have a $[[n, n-2m, 3]]$ quantum code. Indeed, for the corresponding $\overline{S} = \phi^{-1}(C) \leq \overline{E}$, $\overline{S}^\perp \setminus \{0\}$ has minimum q-weight 3, so the code is *non-degenerate*, in the sense defined at the end of Section 6.

Example Continuing our example in the last section, the subspace $\overline{S}$ is spanned by the rows of the matrix

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 1 & | & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & | & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & | & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & | & 1 & 0 & 1 & 1 & 0 \end{bmatrix}.$$

(We have replaced each row x of the generator matrix of $\mathcal{D}_2(4)$ by the two rows ωx and $\overline{\omega} x$ and then applied the inverse of ϕ.) Then $\overline{S}^\perp$ is spanned by $\overline{S}$ and the two additional rows

$$\begin{bmatrix} 0 & 0 & 1 & 1 & 0 & | & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & | & 0 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

If Q is a non-degenerate $[[n, k, d]]$ code and $t = \lfloor (d-1)/2 \rfloor$, then counting independent vectors in $\mathbb{C}^N$ $(N = 2^n)$ gives $\dim Q \cdot |\mathcal{E}| = 2^k[1 + 3(n \text{ choose } 1) + \ldots + 3^t(n \text{ choose } t)] \leq 2^n$ [sorry, don't find how to do binomial coeffs]. The example of the $[[n, n-2m, e]]$ code with $n = (4^m - 1)/3$ gives equality, so it *saturates* the quantum bound.

Appendix: Cryptography and factorisation

In this appendix we give a brief review of the concept of public-key cryptography, the RSA system (its most popular realisation), and the relevance a fast quantum algorithm for factorising large integers would have for this system. No attempts at rigorous definitions of complexity classes or proofs of the assertions will be given.

In any cryptosystem, the *plaintext* to be transmitted is encrypted by some algorithm depending on additional data called the *key*, to produce *ciphertext*. The recipient uses another algorithm to recover the plaintext from the ciphertext and key. The simplest example is the *one-time pad*, the only provably secure cipher system. The plaintext is first encoded as a string of bits of length n, say $a_1a_2\ldots a_n$. The key consists of a string $b_1b_2\ldots b_n$ of n random bits, produced by some physical randomising process such as tossing coins. The encryption algorithm is bitwise addition; so the ciphertext is $c_1c_2\ldots c_n$, where $c_i = a_i + b_i$ (addition mod 2). The decryption algorithm in this case is identical to the encryption algorithm: add the key bitwise (since $c_i + b_i = a_i$). The ciphertext is itself a random string of bits, so an interceptor without knowledge of the key is unable to gain any information. (The security of this system was proved by Shannon [15].)

Note that both the sender and the recipient must have the key, which must be kept secret from the interceptor. The key must either be shared on a previous occasion, or conveyed by a channel which is known to be secure.

Public-key cryptography was invented by Diffie and Hellman [7] in 1975. (In fact, the same idea had been invented six years earlier by James Ellis, an employee of GCHQ, who was unable to publish it because of his employment: see [18].) The idea is that the encryption algorithm and the key are made public, but the decryption is so demanding of computational resources that this knowledge is of no use to the interceptor. However, the recipient (who publishes the key) has some additional information, the *secret key*, which makes the decryption much easier.

More formally, let M be the set of plaintext messages, C the set of ciphertext messages, and K the set of keys. An *encryption system* consists of a pair of functions, *encryption* $e : M \times K \to C$, and *decryption* $d : C \times K \to M$, such that $d(e(m,k),k) = m$ for all $m \in M$ and $k \in K$. A *public-key system* also has a set S of secret keys, and an inverse pair of functions $p : S \to K$ and $q : K \to S$ such that

- computation of $e(m,k)$ and $p(s)$ are easy;
- computation of $d(c,k)$ and $q(k)$ are difficult;
- if $q(k)$ is known, then computation of $d(c,k)$ is easy (in other words, computation of $d(c,p(s))$ is easy).

Each user i of the system selects an element $s_i \in S$, computes $k_i = p(s_i)$, and publishes the result. If user j wants to send a message m to user i, she looks up k_i in the public directory, computes $c = e(m,k_i)$, and transmits this ciphertext. Now i computes $d(m,q(s_i)) = d(m,k_i) = m$. An interceptor knows c and k_i but is faced with the difficult tasks of either computing $d(m,k_i)$ directly or computing $s_i = q(k_i)$.

The RSA system works as follows. Each secret key consists of a pair p, q of large prime numbers (of hundreds of bits), and an integer a coprime to $(p-1)(q-1)$. From this, by Euclid's algorithm, one computes b such that $ab \equiv 1 \pmod{(p-1)(q-1)}$. Now the public key is the pair (N, a), where $N = pq$. The encryption algorithm takes the message m, which is encoded as an integer less than N, and computes the ciphertext $c = m^a \bmod N$. The possessor of the secret key can decrypt this by raising it to the power b mod N, since

$$c^b = m^{ab} \equiv m^1 = m \bmod N.$$

(We use the fact that $\phi(N) = (p-1)(q-1)$, and Fermat's Little Theorem asserting that $m^{\phi(N)} \equiv 1 \pmod{N}$.

Of the interceptor's two strategies, the second (calculating the secret key) involves factorising N, so that b can also be calculated. It can be shown (we omit the details) that the first strategy (decrypting using the public key) is also equivalent to factorising N. An intermediate strategy would be to find b such that $m^{ab} \equiv m \pmod{N}$ for all m. This amounts to finding u such that $m^u \equiv 1 \pmod{N}$, for then b can be found by the Euclidean algorithm. But the smallest such u is the least common multiple of $(p-1)$ and $(q-1)$; knowledge of this determines $(p-1)(q-1)$ and hence p and q.

So the security of the method depends on the assumption that *factorising large numbers is a hard problem.*

The factorisation problem has been studied for centuries, but the best known 'classical' algorithm for it has fractional exponential complexity.

As we mentioned, Shor [17] found a probabilistic algorithm which runs in quantum polynomial time. It depends on the following observation. Suppose that we have computed the order of x mod N, the smallest number t such that $x^t \equiv 1 \pmod{N}$. Suppose that t is even, say $t = 2r$. Then N divides $x^t - 1 = (x^r - 1)(x^r + 1)$. If x^r is not congruent to -1 mod N, then both $x^r - 1$ and $x^r + 1$ have factors in common with N. We can find the g.c.d.s of the pairs $(N, x^r - 1)$ and $(N, x^r + 1)$ by Euclid's algorithm; then we know two different factors of N, and hence the complete factorisation in the RSA case.

The basic idea of quantum computation is to exploit the inherent parallelism of quantum systems. Suppose, for example, that we want to compute 2^n values $f(0), f(1), \ldots, f(2^n - 1)$ of a function simultaneously. We represent the integer i by the binary vector $v \in V = \mathbb{Z}_2^n$ which is its expression in base 2. Now, if n qubit registers are available, we can load them with a superposition of the states e_v, for $v \in V$, as follows: first load each register with 0 (so that we have state e_0); then apply the Hadamard transformation with matrix $(1/\sqrt{2})\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$ (corresponding to a 45° rotation of the 2-dimensional Hilbert space) to each qubit. The resulting state is

$$\frac{1}{\sqrt{2^n}} \sum_{v \in V} e_v.$$

Now suppose that some quantum computation replaces e_v by a state representing $f(i)$, where v represents the integer i. Then, by the linearity of the Schrödinger equation, the same computation replaces the above superposition by a superposition of the values $f(i)$ for $i = 0, \ldots, 2^n - 1$. In the factorisation algorithm, we take $f(i) = x^i \pmod{N}$, where $2^n \geq N$ and x is some chosen integer coprime to N.

The last stage involves finding the period of f from the above superposition, which is done by use of the discrete Fourier transform. The Fourier transform of f is concentrated on multiples of the period, so an observation of the result will with high probability give a small multiple of the period.

Let us take as an example the current objective of quantum computing, the factorisation of 15. We take $x = 2$. Suppose that we have 8 qubit registers arranged in two banks of four, so that states of the system have the form

$$\sum_{v,w \in V} a_{vw}(v \otimes w)$$

in the space $\mathbb{C}^{2^4} \otimes \mathbb{C}^{2^4}$. For convenience we write e_i in place of e_v, where i is the integer in the range $[0, 15]$ whose base 2 representation is v. We begin with the state $e_0 \otimes e_0$ (that is, all registers contain zero). Then by applying a Hadamard transform to each of the first four qubits, we obtain (up to normalisation) $\left(\sum_{i \in [0,15]} e_i\right) \otimes e_0$. The crucial part of the computation replaces $e_i \otimes e_0$ with $e_i \otimes e_{2^i \bmod 15}$. Apart from normalisation, the state is now

$$(e_0 + e_4 + e_8 + e_{12}) \otimes e_1 + (e_1 + e_5 + e_9 + e_{13}) \otimes e_2 + \cdots .$$

We extract the period 4 from the first four registers by a discrete Fourier transform. Now we know that 15 divides $(2^4 - 1) = (2^2 - 1)(2^2 + 1) = 3 \cdot 5$, so the two factors of 15 are $(15, 3) = 3$ and $(15, 5) = 5$.

See Shor [16, 17] for further details.

Physical constraints limit the size of any quantum computer of a type so far proposed to at most about 30 qubits, and also place severe limitations on the number of computational steps which can be carried out. The way round this is to design a fault-tolerant computer, in which errors introduced by the gates and buses are corrected faster than new errors are introduced. This is theoretically possible only if the reliability of the components is above a certain threshold. This threshold has not yet been achieved.

For further reading on quantum computation, see the forthcoming book of Lo *et al.* [11].

References

[1] C. H. Bennett and P. W. Shor, Quantum information theory, *IEEE Trans. Inform. Theory* (1998).

[2] A. R. Calderbank, P. J. Cameron, W. M. Kantor and J. J. Seidel, $\mathbb{Z}_4$-Kerdock codes, orthogonal spreads, and extremal Euclidean line systems, *Proc. London Math. Soc.* (3) **75** (1997), 436–480.

[3] A. R. Calderbank, E. M. Rains, P. W. Shor and N. J. A. Sloane, Quantum error correction via codes over GF(4), *IEEE Trans. Inform. Theory*, **44** (1998), 1369–1387.

[4] A. R. Calderbank, E. M. Rains, P. W. Shor and N. J. A. Sloane, Quantum error correction and orthogonal geometry, *Phys. Rev. Lett.*, **78** (1997), 405–409.

[5] A. R. Calderbank and P. W. Shor, Good quantum error-correcting codes exist, *Phys. Rev. A* **54** (1995), 1098–1106.

[6] R. Cleve and D. Gottesman, Efficient computations of encodings for quantum error correction, *Phys. Rev.* A, **56** (1997) 76–82.

[7] W. Diffie and M. E. Hellman, New directions in cryptography, *IEEE Trans. Inform. Theory* **22** (1976), 644–654.

[8] R. A. Fisher, The theory of confounding in factorial experiments in relation to the theory of groups, *Ann. Eugenics* **11** (1942), 341–353.

[9] R. A. Fisher, A system of confounding for factors with more than two alternatives, giving completely orthogonal cubes and higher powers, *Ann. Eugenics* **12** (1945), 2283–290.

[10] D. Gottesman, Class of quantum error-correcting codes saturating the quantum Hamming bound, *Phys. Rev.* A, **54** (1996), 1862–1868.

[11] H.-K. Lo, S. Popescu and T. P. Spiller (eds.), *Introduction to Quantum Computation*, in press.

[12] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland Publishing Co., Amsterdam, 1977.

[13] J. Preskill, Lecture notes for Physics 229 at Caltech, `http://theory.caltech.edu/~preskill/ph229`.

[14] R. L. Rivest, A. Shamir and L. Adelman, A method of obtaining digital signatures and public-key cryptosystems, *Comm. Assoc. Comput. Mach.* **21** (1978), 120–126.

[15] C. E. Shannon, Communication theory of secrecy systems, *Bell Systems Tech. J.* **28** (1949), 657–715.

[16] P. W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, *SIAM J. Computing* **26** (1997), 1484–1509.

[17] P. W. Shor, Quantum Computing, ICM 1998 lecture, `http://www.research.att.com/~shor/papers/ICM.ps`.

[18] P. Wayner, British document outlines early encryption discovery, *New York Times*, 24 December 1997.

[19] D. Welsh, *Codes and Cryptography*, Oxford University Press, Oxford, 1988.

[20] W. H. Zurek, Decoherence and the transition from quantum to classical, *Physics Today* **44** (1991), 36–44.