Proof of Claim 14.20. Suppose not. Then for some $v, v' \in A$ we have $u + v = u' + v'$, and hence, $v + v' = u + u'$. Let c, c' be the vectors from C for which $u = c_1 v_1 + \cdots + c_k v_k$ and $u' = c'_1 v_1 + \cdots + c'_k v_k$. Then

$$v + v' = u + u' = (c_1 + c'_1)v_1 + (c_2 + c'_2)v_2 + \cdots + (c_k + c'_k)v_k.$$

Since vectors c and c' differ in at least three coordinates, we have on the right-hand side the sum of at least three vectors, say $v_{i_1} + \cdots + v_{i_l}$, with $l \geqslant 3$. But then in the equation (14.6) we can replace these three (or more) vectors $v_{i_1}, \ldots, v_{i_l}$ by two vectors v, v', which contradicts the minimality of k. □

The same argument also implies that no two distinct vectors $c, c' \in C$ can lead to one and the same vector $u \in B$, which means that $|B| = |C|$.

This, together with Claim 14.20, implies

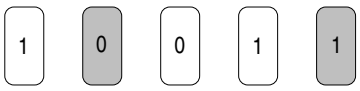
$$|A| \cdot |C| = |A| \cdot |B| = \sum_{u \in B} |u + A| = \left| \bigcup_{u \in B} (u + A) \right| \leqslant |\operatorname{span} A| .$$

Hence, $\log_2 |C| \leqslant \log_2(1/\alpha)$ which, together with Claim 14.19, yields the desired upper bound (14.5) on k. □

14.5 The flipping cards game

There are situations in theory of computing, where switching to the linear algebra *language* alone can lead to interesting results. In particular, linear combination and/or scalar product can often be used to encode some useful information about the input vectors which, in its turn, can lead to surprisingly efficient algorithms.

Suppose that we have two 0-1 vectors $u = (u_1, \ldots, u_n)$ and $v = (v_1, \ldots, v_n)$ of length n. We want to decide whether $u = v$, but our access to the bits is very limited: at any moment we can see at most one bit of each pair of the bits u_i and v_i. We can imagine the corresponding bits to be written on two sides of a card, so that we can see all the cards, but only one side of each card:



A *probe* consists in flipping of one or more of the cards. After every probe we can write down some information but the memory is not reusable – after the next probe we have to use new memory (i.e., we cannot wipe it out). Moreover, this is the only memory for us: seeing the information written here (but not the cards themselves), we ask to flip some of the cards; seeing the actual values of the cards and using the current information from the memory,

we either give an answer or we write some additional bits of information in the memory; after that the cards are closed for us, and we make the next probe.

Suppose we are charged for every bit of memory that we use but not for the number of probes. The goal is to decide if both sides of all cards are the same using as little of memory as possible. Of course, n bits of memory are always enough: simply write u in the memory, and flip all the cards to see v. Can we do better? To enjoy the next two results, the reader is invited to stop for a moment and try to imagine a protocol which uses less than n bits of memory.

Theorem 14.21. *Let $n = r^2$ for some $r \geqslant 1$. It is possible to test the equality of two vectors in $\{0,1\}^n$ using only $r+1$ probes and writing down only r bits in the memory.*

Proof. The following protocol is due to J. Edmonds and R. Impagliazzo. Split the given vectors u and v into r pieces of length r: $u = (u^1, \dots, u^r)$ and $v = (v^1, \dots, v^r)$. In the first probe look at vector u and compute the vector

$$w_0 \rightleftharpoons u^1 + u^2 + \cdots + u^r,$$

where the sum is over $\mathbb{F}_2$. Write down this vector w_0 in the memory (using r bits), and make subsequent r probes as follows. During the ith probe flip the cards of the ith piece; compute the vector

$$w_i \rightleftharpoons u^1 + \cdots + u^{i-1} + v^i + u^{i+1} + \cdots + u^r$$

and just test if the obtained vector w_i coincides with the vector w_0 (written in the memory). Answer "$u = v$" if all the vectors $w_1, \dots, w_r$ coincide with w_0, and "$u \neq v$" otherwise. If we answer "$u = v$", we know that, after the first probe, $u^1 + u^2 + \cdots + u^r = v^1 + u^2 + \cdots + u^r$ and hence $u^1 = v^1$; the same argument is valid for other probes, hence $u = v$ and the protocol is correct. □

Using the language of scalar products, Pudlák and Sgall (1997) have shown that, in fact, $O((\log n)^2)$ bits are enough.

Theorem 14.22. *It is possible to test the equality of two vectors in $\{0,1\}^n$ using only $O(\log n)$ probes and writing down only $O(\log n)$ bits in the memory about each probe.*

Proof. Each probe corresponds to a subset $I \subseteq \{1, \dots, n\}$; after this probe we see n bits: $|I|$ bits $\{u_i : i \in I\}$ of u and $n - |I|$ bits $\{v_i : i \notin I\}$ of v. We think of u and v as 0-1 vectors in real vector space $\mathbb{R}^n$. The idea is to compute (a square of) the Euclidean distance

$$\begin{aligned} \|u - v\|^2 &= \langle u, u \rangle + \langle v, v \rangle - 2 \langle u, v \rangle \\ &= \sum_{i=1}^{n} u_i^2 + \sum_{i=1}^{n} v_i^2 - 2 \left[\left(\sum_{i=1}^{n} u_i \right) \cdot \left(\sum_{i=1}^{n} v_i \right) - \sum_{i \neq j} u_i v_j \right] \end{aligned}$$

of u and v, and check if it is 0. We compute $\langle u,u\rangle$ and $\langle v,v\rangle$ each using one probe (probe $I=\{1,\dots,n\}$ for $\langle u,u\rangle$ and probe $I=\emptyset$ for $\langle v,v\rangle$) and $\lceil\log(n+1)\rceil$ bits of memory (to write down these two numbers between 0 and n). It remains to compute the product $\langle u,v\rangle=\sum_{i=1}^n u_iv_i$.

To do this, we first compute the product $N=(\sum_{i=1}^n u_i)\,(\sum_{i=1}^n v_i)$ using the same probes and additional $2\lceil\log(n+1)\rceil$ bits of the memory (to write the value of this product which lies between 0 and n^2). To compute the desired product $\langle u,v\rangle$ we need to subtract from N the sum $\sum_{i\neq j}u_iv_j$ of cross-terms. This is easily done using $2\lceil\log n\rceil$ probes: choose them so that each of the cross-terms can be computed by one of them, and for each probe sum all these terms assigned to it. After each of these probes we write the resulting partial sum using $O(\log n)$ bits of memory. □

Exercises

14.1.$^-$ Prove the Pythagoras theorem: if the vectors u,v are orthogonal, then $\|u+v\|^2=\|u\|^2+\|v\|^2$.

14.2.$^-$ Show that the minimal distance of a linear code coincides with the minimum weight of its non-zero vector. *Hint*: Every linear code contains the zero vector.

14.3. Prove the following stronger version of Proposition 14.17. Let C be a linear code of length n and minimal distance $k+1$ and let $C^\perp$ be its dual. Then for every subset S of $l\leqslant k$ coordinates, every 0-1 string of length l appears as a projection of $C^\perp$ onto S one and the same number of times.

Hint: Take a matrix whose rows form a basis of $C^\perp$, observe that every k columns of this matrix are linearly independent and use Proposition 14.3.

14.4.$^-$ Let $V\subseteq\mathbb{F}_2^n$ be a subspace of dimension d. Show that $|V|=2^d$.

14.5.$^-$ Let $\mathcal{F}$ be a family of subsets of an n-element set such that: (i) every set of $\mathcal{F}$ has an *even* number of elements, and (ii) each pair of sets share an *even* number of elements. Construct such a family with at least $2^{\lfloor n/2\rfloor}$ sets.

14.6 (Babai–Frankl 1992). Show that the upper bound $2^{\lfloor n/2\rfloor}$ in the previous exercise cannot be improved.

Hint: Let S be the set of incidence vectors of all sets in $\mathcal{F}$, and let U the span of this set (over $\mathbb{F}_2$). Argue that the rules (i) and (ii) imply that U is a subspace of $U^\perp$, and apply Proposition 14.2.

14.7.$^{(!)}$ Prove the following "Oddtown Theorem" (see Babai and Frankl (1992) for the explanation of this name). Let $\mathcal{F}$ be a family of subsets of an n-element set such that: (i) every set of $\mathcal{F}$ has an *odd* number of elements, and (ii) each pair of sets share an *even* number of elements. Prove that then $|\mathcal{F}|\leqslant n$. Compare this with Exercise 14.5.

Hint: The incidence vectors of sets in $\mathcal{F}$ are linear independent over $\mathbb{F}_2$.