Acta Cybernetica **16** (2004) 443–449.

On Computing the Hamming Distance

Gerzson Kéri* and Ákos Kisvölcsey†

Abstract

Methods for the fast computation of Hamming distance developed for the case of large number of pairs of words are presented and discussed in the paper. The connection of this subject to some questions about intersecting sets and Hadamard designs is also considered.

Keywords: covering radius, Hamming distance, Hamming weight, intersecting sets, minimum distance.

1 Introduction and notation

Let Z_q^n denote the set of all n-tuples $(x_1, x_2, \ldots, x_n)$, where $Z_q = \{0, 1, \ldots, q-1\}$. The elements of the set Z_q^n are called words, and the Hamming distance $d(x, y)$ between two words $x, y \in Z_q^n$ is defined as the number of coordinates in which they differ.

One may encounter the problem of determining the Hamming distance for a large number of pairs of words in the same space. This is, for example, the case when the minimum distance or the covering radius for a lot of codes $C_i \subseteq Z_q^n$ are to be determined. (See also Section 6.) The Hamming distance and Hamming weight find many applications also in cryptography [5]. For problems like this there emerges the need for faster computation.

In the paper a general method is presented and discussed for the fast computation of the Hamming distance. This method is related to a problem of intersecting sets.

We emphasize that the suggested (and applied) method is not faster than the direct method if the Hamming distance is to be determined for only a small number of pairs of words. It is proposed for application only if the number of pairs is large enough.

The notation & is used for the bitwise "and" operation, XOR for the bitwise "exclusive or" operation. The wgt function counts the number of 1-s in a binary

*Computer and Automation Research Institute, Hungarian Academy of Sciences, H-1111 Budapest Kende u. 13-17, Hungary, e-mail: keri@sztaki.hu
Supported in part by the Hungarian National Research Fund OTKA, Grant No. T043276.
†Alfréd Rényi Institute of Mathematics, Hungarian Academy of Sciences, H-1053 Budapest Reáltanoda u. 13–15, Hungary, e-mail: ksvlcs@renyi.hu

integer; it can be given by formula as

$$\mathrm{wgt}(a) = \sum_{k=0}^{\infty} \left(\lfloor a/2^k \rfloor \pmod 2 \right).$$

The symmetric difference of two sets is denoted by $\triangle$:

$$A \triangle B = (A \cap \overline{B}) \cup (\overline{A} \cap B).$$

2 Hamming distance of q-ary vectors and q-ary distance of integers

Clearly, there is a one-to-one correspondence between a word $x = (x_1, x_2, \ldots, x_s) \in Z_q^s$ and a nonnegative integer n in the interval $0 \le n \le q^s - 1$:

$$x \longleftrightarrow n = \sum_{i=1}^{s} x_i q^{s-i}.$$

We define the q-ary distance $d_q(a, b)$ of two nonnegative integers as the Hamming distance of the corresponding words in any space Z_q^s where

$$s \ge \max\left(\log_q(a+1), \log_q(b+1) \right).$$

We look for a fast way of computing the Hamming distance of words, stored in the form of q-ary integers for a large number of pairs of words in the same space. That means the computing of $d_q(a, b)$ for pairs of integers (a, b). This problem arises, for example, when the minimum distance or the covering radius of many codes are to be checked.

The minimum distance of a code $C \subseteq Z_q^s$ is defined as

$$\min\{d(x, y) \mid x, y \in C,\ x \ne y\}.$$

The covering radius of a code $C \subseteq Z_q^s$ is the smallest positive integer R such that for an arbitrary $x \in Z_q^s$, there exists one (or more) $y \in C$ with $d(x, y) \le R$. In other words,

$$R = \max\{d(x, C) \mid x \in Z_q^s\},$$

where

$$d(x, C) = \min\{d(x, y) \mid y \in C\}.$$

3 The binary case ($q = 2$)

Fast methods to calculate Hamming distances (and Hamming weights) in the binary case are known from the literature, see e.g. [5] where the theme is discussed within a more general context. There can be found many communications as well as computer codes related to the subject also on the web.

Here, we describe in short the substance of the method as follows.

For $q = 2$, i. e. for binary numbers, clearly

$$d_2(a, b) = \mathrm{wgt}(a \text{ XOR } b).$$

This fact suggests arranging the weights into an array consisting of the array elements

$$\mathrm{wgt}(1), \quad \mathrm{wgt}(2), \quad \ldots, \quad \mathrm{wgt}(2^L - 1),$$

where the exponent L depends on the computational environment (available hardware and software, programming language etc.).

The same method can be applied with a slight modification also for numbers greater than $2^L - 1$ if we split them into 2 or more parts. If, e.g., $n > 2^L - 1$ but $n \leq 2^{2L} - 1$, then – referring to the identity

$$\mathrm{wgt}(n) = \mathrm{wgt}\left(\lfloor n/(2^L)\rfloor\right) + \mathrm{wgt}\left(n \,(\mathrm{mod}\ 2^L)\right),$$

– we can use the formula

$$d_2(a, b) = \mathrm{wgt}\left(\lfloor (a \text{ XOR } b)/(2^L)\rfloor\right) + \mathrm{wgt}\left((a \text{ XOR } b) \,(\mathrm{mod}\ 2^L)\right).$$

That way, an array of length 2^L is enough for treating integers as large as we want.

Note that the division by 2^L can be performed simply by a right shift of the dividend.

4 Method for the case $q > 2$

When $q > 2$, the q-ary distance $d_q(a, b)$ of two integers cannot be determined immediately by the help of the weight function. What can be done is to have a and b mapped to (longer) integers A and B such that

$$d_2(A, B) = k \cdot d_q(a, b) \quad \text{for any } a, b \in Z_q^s,$$

where k is a positive integer, depending only on the value of q and the mapping.

For this purpose, let

$$\varphi_q : Z_q \longrightarrow Z_2^t$$

with an appropriate t, a mapping having the property of

$$\mathrm{wgt}\left(\varphi_q(\alpha) \text{ XOR } \varphi_q(\beta)\right) = k \tag{1}$$

for any pair $\alpha, \beta \in Z_q$, $\alpha \neq \beta$ for a positive integer k.

Clearly, φ_q generates a mapping of Z_q^s to Z_2^{st}, if we apply φ_q to all q-ary digits of $n \leq q^L - 1$. The corresponding mapping for q-ary integers can be written by the formula

$$\Phi_q(n) = \sum_{j=0}^{L-1} 2^{jt} \cdot \varphi_q\left(\lfloor n/q^j \rfloor \pmod q\right).$$

Now, for any $a, b \leq q^L - 1$, $\operatorname{wgt}(\varphi_q(\alpha) \text{ XOR } \varphi_q(\beta)) = k$ implies $\operatorname{wgt}(\Phi_q(a) \text{ XOR } \Phi_q(b)) = k \cdot d_q(a, b)$.

From the point of view of effectiveness, the value of t should be kept as small as possible.

The same problem can be translated to a problem with intersecting sets. For this purpose, consider a set S consisting of t elements:

$$S = \{u_1, u_2, \ldots, u_t\}.$$

Consider also the binary representation of $\varphi_q(\alpha)$ as

$$\varphi_q(\alpha) = (b_1(\alpha), b_2(\alpha), \ldots, b_t(\alpha))$$

for any $\alpha \in Z_q$, $\varphi_q(\alpha) : Z_q \longrightarrow Z_2^t$.

Define the subsets $S_1, S_2, \ldots, S_q$ of S as follows:

$$u_i \in S_{\alpha+1} \text{ if and only if } b_i(\alpha) = 1.$$

To find a mapping $\varphi_q(\alpha)$ having the property (1) is equivalent to find a set S and q subsets $S_1, S_2, \ldots, S_q \subseteq S$ such that the cardinality of the symmetric differences

$$S_i \triangle S_j = (S_i \cap \overline{S_j}) \cup (\overline{S_i} \cap S_j)$$

is constant for any pairs of S_i and S_j, provided $i \neq j$, where $\overline{S_i}$ is used for $S \setminus S_i \quad (i = 1, 2, \ldots, q)$.

For the system of sets $S_1, S_2, \ldots, S_q$ with the property described above, the following notices can be taken.

1. Consider the sets

$$U_i = S_1 \triangle S_{i+1}$$

for $i = 0, \ldots, q-1$. Now, we have $U_0 = \emptyset$, and

$$|U_i| = k$$

for $i = 1, \ldots, q-1$. It is easy to see that $U_i \triangle U_j = S_i \triangle S_j$, thus also $|U_i \triangle U_j| = k$ holds. Clearly, $|U_i \triangle U_j| = |U_i| + |U_j| - 2|U_i \cap U_j|$, consequently,

$$|U_i \cap U_j| = \frac{k}{2}$$

for every $i, j \geq 1$, $i \neq j$. From this, it also follows that k must be even. So, we have a k-uniform family $U_1, \ldots, U_{q-1}$ on the t-element ground set S, such that any pair

of sets shares the same number of elements. By using linear algebraic methods, Bose [2] proved that $t \geq q-1$ for such set-systems. Later in the paper we show that this bound can be achieved in some cases (cf. Examples 2, 3).

2. Assume now that $t = q-1$. Ryser [7] showed that in this case every point in S is contained in exactly k sets from $U_1, \ldots, U_{q-1}$. By doubly counting the triplets (u, U_i, U_j), where $u \in U_i \cap U_j,\ i \neq j$, we get

$$t\binom{k}{2} = \binom{q-1}{2}\frac{k}{2}.$$

From this, we obtain $q = 2k$. Since k is even, if q is not divisible by 4, then $t \geq q$ must hold. Obviously, this bound can be achieved in any case (cf. Example 1).

3. Suppose that q is divisible by 4. Let $q = 4\lambda$, $k = 2\lambda$, where λ is a positive integer. What we want to find is a symmetric block design $S_\lambda(2, 2\lambda, 4\lambda-1)$, that is, a 2λ-uniform set-system $U_1, \ldots, U_{4\lambda-1}$ on a $t = 4\lambda - 1$-element ground set S, such that every pair of sets has an intersection of size λ. If we take the complement sets $V_i = S \setminus U_i$, then

$$|V_i| = 2\lambda - 1,$$

and $V_i \cap V_j = S \setminus (U_i \cup U_j)$. Since $|U_i \cup U_j| = |U_i| + |U_j| - |U_i \cap U_j| = 3\lambda$, we have

$$|V_i \cap V_j| = \lambda - 1.$$

So, equivalently, we want to find a so-called *Hadamard design* $S_{\lambda-1}(2, 2\lambda-1, 4\lambda-1)$. It is known that such a system exists if and only if there is a Hadamard matrix of order 4λ. An *Hadamard matrix of order* m is an $m \times m$ matrix H with entries $\{1, -1\}$ such that its row vectors are orthogonal to each other, as well as its column vectors, i.e., $HH^T = H^T H = mI$. It is conjectured that there is an Hadamard matrix of order 4λ for every positive integer λ, and thus, we can have $t = q-1$.

5 Examples

1. For arbitrary $q > 2$, we may choose $t = q$ and $\varphi_q(\alpha) = 2^\alpha$.
   Then, wgt $(\varphi_q(\alpha)$ XOR $\varphi_q(\beta)) = 2$ for $\alpha \neq \beta$.
   In the terminology of intersecting sets

$$S = \{u_1, u_2, u_3\},\ S_1 = \{u_1\},\ S_2 = \{u_2\},\ S_3 = \{u_3\}.$$

2. For $q = 4$, let $t = 3$ and $\varphi_4(\alpha) = 0, 3, 5, 6$ for $\alpha = 0, 1, 2, 3$, respectively.
   Now, wgt $(\varphi_4(\alpha)$ XOR $\varphi_4(\beta)) = 2$ again for $\alpha \neq \beta$.
   In the terminology of intersecting sets

$$S = \{u_1, u_2, u_3\},\ S_1 = \emptyset,\ S_2 = \{u_1, u_2\},\ S_3 = \{u_1, u_3\},\ S_4 = \{u_2, u_3\}.$$

3. For $q = 2^{m+1}$, $m \geq 1$, the following recursion can be applied:

$$\varphi_{2^{m+1}}(2\alpha - 1) = (2^{2^m} + 1) \cdot \varphi_{2^m}(\alpha),$$

$$\varphi_{2^{m+1}}(2\alpha) = (2^{2^m} - 1) \cdot (\varphi_{2^m}(\alpha) + 1)\,.$$

In this case $t = q - 1 = 2^{m+1} - 1$ can be specified. The inequality

$$\varphi_{2^{m+1}}(\alpha) \le 2^{2^{m+1}-1} - 1 \quad \text{for} \quad 0 \le \alpha \le 2^{m+1} - 1$$

can be proved by induction. The multiplier k assumes the value 2^m.

6 Application of the method for checking the covering radius of codes

The methods described in the paper found an application in [4] for computing the covering radii of a huge number of codes. This computation resulted in the improvement of known lower bounds on the covering radii for several families of codes. This way, general inequalities (sometimes equalities) were found for the covering radii of an infinite number of codes; however, to obtain these results, a finite (but very large) number of codes had to be considered and the covering radii of more than 150 million codes were checked by using a computer.

This job could not have been completed within a reasonable time by applying the direct method for the computation of the Hamming distance, i. e. by counting the number of non-identical coordinates.

By using the weight function and the "exclusive or" operation, the check of binary codes was completed 6–8 times faster than by the direct method. For ternary and mixed ternary/binary codes, using the mapping φ and applying the weight function for the transformed vectors resulted in an additional gain in the CPU time. Thus, finally, the whole job of checking the covering radii of millions of codes required about 30 days of CPU time (instead of 300 days or more, which would have been required by applying the direct method).

Finally, we summarize the computational aspects of the method applied for the case of a mixed ternary/binary Hamming space. The process of the method needs three initial steps as follows:

1. We start with storing in two arrays the powers of 2 and 3 for exponents $0, 1, \ldots$ until these can be represented as long integers (arrays pow2 and pow3).

2. The weights of binary integers are stored in another array wgt of long integers:

$$\mathrm{wgt}(n) = \sum_{j \ge 0} \mathrm{sign}\,(n \;\&\; \mathrm{pow2}(j))\,.$$

3. The values of $\Phi_3(n)$ are stored also in an array of long integers:

$$\Phi_3(n) = \sum_{k=0}^{L-1} 2^{3k\,+\,(\lfloor n/3^k \rfloor)\,(\mathrm{mod}\ 3)}.$$

After these steps of initialization, the computation of Hamming distances is done as follows.

For arbitrary words x, y of the mixed Hamming space $Z_3^{n_1} \oplus Z_2^{n_2}$, these words can be given as pairs consisting of a ternary and a binary integer:

$$x = (x_t,\ x_b),\ y = (y_t,\ y_b).$$

Then, the Hamming distance $d_{3,2}(x, y)$ is computed by using the formula

$$d_{3,2}(x, y) = \frac{\mathrm{wgt}(\Phi(x_t)\ \mathrm{XOR}\ \Phi(y_t))}{2} + \mathrm{wgt}(x_b\ \mathrm{XOR}\ y_b).$$

Acknowledgement

The authors are grateful to Patric R. J. Östergård for his helpful comments and suggestions. The first author would like to thank the Hungarian National Research Fund (OTKA) for partial financial support (Grant No. T043276).

References

[1] I. Anderson, *Combinatorics of finite sets*, The Clarendon Press, Oxford University Press, New York (1987).

[2] R. C. Bose, A note on Fisher's inequality for balanced incomplete block designs, *Ann. Math. Statistics*, Vol. 20 (1949) 619–620.

[3] G. Cohen, I. Honkala, S. Litsyn and A. Lobstein, *Covering Codes*, North-Holland, Amsterdam (1997).

[4] G. Kéri and P. R. J. Östergård, Further results on the covering radius of small codes, submitted for publication.

[5] H. Lipmaa and S. Moriai, Efficient Algorithms for Computing Differential Properties of Addition, *Fast Software Encryption '2001 (M. Matsui, ed.), Lecture Notes in Computer Science* Vol 2355., Springer-Verlag (2002), 336–350.

[6] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland, Amsterdam (1977).

[7] H. J. Ryser, A note on a combinatorial problem, *Proc. Amer. Math. Soc*, Vol. 1 (1950) 422–424.

Received January, 2004