Matrix-Vector Multiplication via Erasure Decoding

Peter Trifonov
Saint-Petersburg State Polytechnic University
Distributed Computing and Networking Department
194021. Polytechnicheskaya 21, room 104, Saint-Petersburg, Russia
Email: petert@dcn.infos.ru

Abstract

The problem of fast evaluation of a matrix-vector product over $GF(2)$ is considered. The problem is reduced to erasure decoding of a linear error-correcting code. A large set of redundant parity check equations for this code is generated. The multiplication algorithm is derived by tracking the execution of the message-passing algorithm on the obtained set of parity check equations. The obtained algorithms outperform the classical ones.

I. Introduction

Let us consider the problem of efficient evaluation of a matrix-vector product $y = Ax$, where $x \in GF(2^p)^k, p > 0$, is an arbitrary vector, and $A \in GF(2)^{r \times k}$ is a fixed matrix. This task is equivalent to computing a number of modulo-2 sums of some elements of x vector. Such problem frequently arises in coding theory, cryptography, computer algebra and many other areas of computer science. Straightforward evaluation of the product requires $rk/2$ operations in average. However, in many cases more efficient algorithms are needed. Such algorithms can be derived easily if matrix A possesses some algebraic structure [1], [2], [3]. However, sometimes the algebraic structure of the matrix is not apparent or is difficult to exploit. In this case it may be possible to employ the "Four Russians algorithm" (V.L. Arlazarov, E.A. Diniz, M.A. Kronrod, I.A. Faradjev) [4] for Boolean matrix multiplication, which requires $2k^2/\log_2 k$ summations over $GF(2^m)$. Alternatively, one could employ common subexpression elimination techniques, which are actively used in modern optimizing compilers [5]. However, the existing CSE algorithms are not able to exploit fully the algebraic properties of the underlying exclusive-OR operation.

In this paper we propose a method for systematic derivation of fast matrix-vector multiplication algorithms for arbitrary binary matrices A. The method is based on the concept of erasure decoding and message-passing algorithm. The paper is organized as follows. The fast algorithm derivation method is described in Section II. Numeric results are presented in Section III. Finally, some conclusions are drawn.

II. Automatic derivation of a matrix-vector multiplication algorithm

A. Message-passing algorithm

Low-density parity check codes and message-passing decoding algorithm have received great attention during last ten years due to their strong error correction capability and low decoding complexity [6]. Recall, that the message-passing algorithm operates on a bipartite graph (Tanner graph). Check nodes of this graph correspond to the parity check equations of a given linear code, while the variable nodes correspond to the codeword symbols (or variables used in the check equations). Check node i is connected to variable node j if and only if variable i is used in the j-th check equation. In the case of erasure decoding, codeword symbols (and corresponding variable nodes in the Tanner graph) may be either known or unknown. By active node we denote a check node such that all its neighbours except one are known. Message passing decoding effectively consists in finding an active check node, expressing the unknown codeword symbol via the known ones using the corresponding check equation, and marking the recovered symbol as known. The complexity of this algorithm is proportional to the number of edges in the Tanner graph, i.e. density of the check matrix used to construct the Tanner graph. It becomes particularly efficient if this matrix is sparse.

It must be recognized that the sparse system of parity check equations is not a property of a specific code, but just one of many possible representations of this linear code. For any given linear code it is always possible to derive a system of sparse parity check equations, completely describing the code. The sparseness of the obtained check matrix depends on the weight spectrum of the dual code. However, one should keep in mind that the performance of the message passing algorithm depends strongly on the degree distribution of the assoicated Tanner graph and stopping set structure [7], [8], [9]. Each parity check matrix is associated with the minimum stopping distance d_s, which gives the number of erasures correctable by the message passing algorithm operating on the associated Tanner graph. Minimum stopping distance can not exceed the minimum distance of the code d_c, and can be substantially smaller than it, for poorly constructed check matrices. However, it is always possible to make $d_s = d_c$ by introducing sufficient amount of redundant check equations [10].

B. Multiplication via decoding

The problem of computing $y = Ax$ is equivalent to finding a solution of the homogenous system of linear equations

$$\begin{pmatrix} I & A \end{pmatrix} \begin{pmatrix} y \\ x \end{pmatrix} = 0, \tag{1}$$

such that the last k entries of the solution vector are given by x. This can be also considered as recovering first r erased symbols in a codeword of a linear binary code $\mathcal{C}$ with check matrix $H = \begin{pmatrix} I & A \end{pmatrix}$. Hence, any fast erasure decoding algorithm for this code can be used to constuct a fast algorithm for multiplication by A. In principle, one can immediately employ the message-passing decoding algorithm to solve (1). However, this would be equivalent to straightforward computation of $y = Ax$. The main idea of the proposed method is to transform the check matrix in such a way that the message passing algorithm becomes more efficient than straightforward matrix-vector multiplication.

Het $H = \begin{pmatrix} I & A \end{pmatrix}$ be the original check matrix of code $\mathcal{C}$, which can be also considered as a generator matrix of the dual code $\mathcal{C}^\perp$. Let $\{h_1, \ldots, h_M\}$ be a set of $\mathcal{C}^\perp$ codewords such that $\mathrm{wt}(h_i) \leq w$, where w is a sufficiently small value being a parameter of the algorithm. Let us construct an extended parity check matrix $\widehat{H}$, which contains all rows of H and vectors h_i. This matrix includes H as a submatrix, so the message-passing algorithm is guaranteed to complete successfully. Indeed, one can always compute $y_i = A_{i,*}x, 1 \leq i \leq r$. However, the additional rows of the extended matrix $\widehat{H}$ correspond to auxiliary identities, which can be used to express some components of vector y via those computed earlier. By construction, the weight of the additional rows does not exceed w, i.e. the corresponding identities allow recovery of codeword symbols with at most $w - 1$ operations, while the identities corresponding to the original matrix H require $k/2$ operations in average. Observe that exactly r identities out of $r + M$ would be actually used by the message passing algorithm.

Let $\widetilde{H}$ be a matrix consisting of the rows corresponding to the parity check identities actually used by the message-passing algorithm. The decoding complexity can be further reduced if one eliminates common subexpressions arising in these identities. Let $n_0 = r + k, \widetilde{H}_0 = \widetilde{H}, r_0 = r, i = 1$. Let us find a pair $(u, v), 1 \leq u < v \leq n_{i-1}$ of columns in $\widetilde{H}_{i-1}$ such that 1's in these columns occur simultaneously in more than one row. Let $j_1, j_2, \ldots$ be the numbers of these rows. Let $r_i = r_{i-1} + 1, n_i = n_{i-1} + 1$. Construct matrix $\widetilde{H}_i$ by appending to $\widetilde{H}_{i-1}$ an additional weight-3 row with 1's in positions u, v and n_{i-1}, and adding this additional row to rows $j_1, j_2, \ldots$. This would effectively eliminate common subexpression $c_u + c_v$ in the corresponding parity check identities. If there exist other common subexpressions, let $i = i + 1$ and repeat the above transformations. It can be seen that the described transformations lead to a check matrix $\overline{H}$ with orthogonal rows. This implies that the associated Tanner graph is free of length-4 loops. This approach was independently proposed in [11].

For example, the described algorithm transforms the matrix $\tilde{H}_0 = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$ into

$$\tilde{H}_1 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \text{ and } \overline{H} = \tilde{H}_2 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

Observe that the sequence of check matrices $\tilde{H}_i$ corresponds to a set of codes with parameters $(n, k), (n+1, k), \ldots, (n+i, k), \ldots$. All codewords $(c_1, c_2, \ldots, c_{n+i})$ of the i-th code in this sequence contain a length-n prefix being a codeword of the original code $\mathcal{C}$. The remaining part corresponds to auxiliary variables used in the matrix-vector multiplication algorithm being designed.

Having obtained a set of orthogonal parity checks, one can generate a sequence of operations needed to recover the codeword by tracking the execution of the message passing algorithm. Observe that this has to be done only once at the design stage. The number of operations in the obtained multiplication algorithm can be computed as

$$f = \sum_{j=1}^{\overline{r}} \operatorname{wt} \overline{H}_{j,*} - \overline{r}, \tag{2}$$

where $\overline{r}$ is the number of rows in matrix $\overline{H}$, and $\overline{H}_{j,*}$ is the j-th row of this matrix.

C. Optimizing the multiplication algorithm

In general, there are many different ways to construct $\tilde{H}$ matrix, leading to the algorithms with different complexity. At each step of the message passing algorithm one can select different parity check identities to be used for computing a new component of y, and even different components of y to be computed. The decision made at each step affects not only the subsequent steps, but also the structure of common subexpressions, as described above. This leads to a huge number of alternatives to be checked while constructing the fast matrix-vector multiplication algorithm. Therefore we propose a randomized method, which is not guaranteed to find an optimal algorithm, but provides quite good results in practice.

1) Construct matrix $H := \begin{pmatrix} I & A \end{pmatrix}$. Let $F := \infty, s := 1$.
2) Find at most P vectors $h_i : \operatorname{wt} h_i \leq w$ in the space of rows of H matrix. For example, this can be done using the algorithm in [12]. Construct matrix $\widehat{H}$ and the associated Tanner graph. Mark r first variable nodes of the Tanner graph as unknown, and the remaining k ones as known.
3) Let $\tilde{H}$ be a $0 \times (r + k)$ matrix.
4) Let $\mathcal{B}$ be the set of active check nodes in the Tanner graph. Let $\mathcal{B}_l \subset \mathcal{B}$ be the set containing at most l active nodes corresponding to rows in $\widehat{H}$ with the smallest weight.
5) Select randomly $j \in \mathcal{B}_l$, and let i be the only unknown variable node connected to check node j. Append the row corresponding to check node j to matrix $\tilde{H}$, and mark variable node i as known.
6) If there exists any unknown node, go to step 4.
7) Apply the common subexpression elimination algorithm described above to matrix $\tilde{H}$. Let f be the number of operations in the corresponding multiplication algorithm.
8) If $f < F$, let $F := f$ and keep the obtained sequence of operations.
9) Let $s := s + 1$. If $s < N$, go to step 3.

The described algorithm has parameters w, P, l, and N, which denote the maximal weight of the additional rows in the extended parity check matrix, their maximal number, the number of different evaluation paths

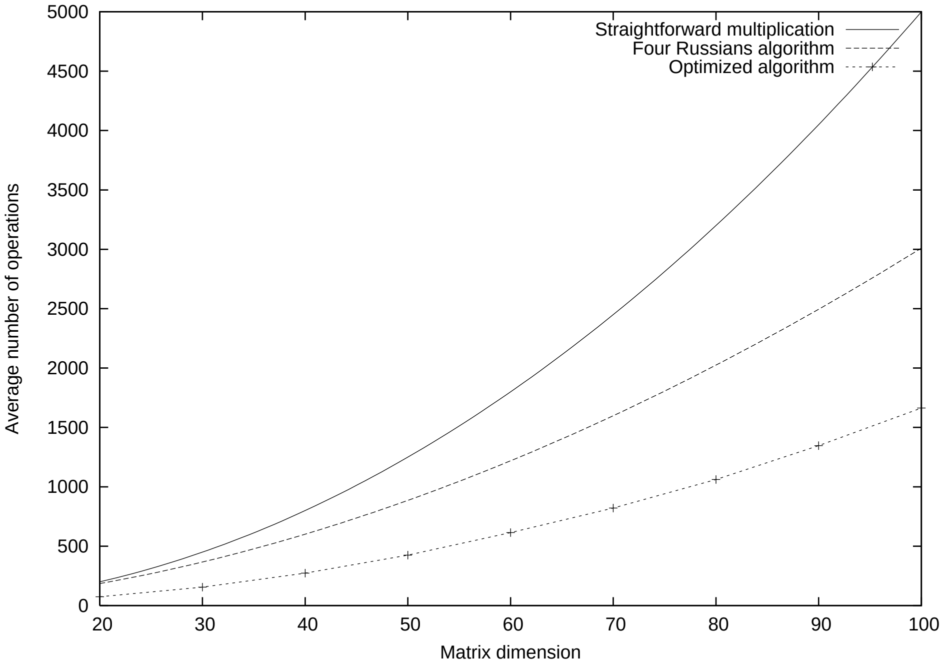

Fig. 1. Complexity of matrix-vector multiplication

considered at each step of the algorithm, and the number of attempts to be performed, respectively. Obviously, increasing N allows one to obtain more efficient multiplication algorithms. The optimal values of w and l depend strongly on the matrix A and have to be optimized interactively. Increasing the value of w increases the number of active nodes available at each iteration of the algorithm, but increases also the number of operations needed to recover the unknown codeword symbols using the corresponding parity check equations. Increasing l extends the search space of the optimization algorithm, enabling potentially more efficient solutions to be found, but increases also the time needed.

III. Numeric results

Figure 1 presents a complexity comparison of different matrix-vector multiplication algorithms. 7 $k \times k$ random matrices were generated for $k = 20..80$, and the method described above was employed to derive a multiplication algorithm. Additionally, the curves corresponding to the complexity of the straightforward and "Four Russians'" algorithms ($k^2/2$ and $2k^2/\log_2 k$ summations, respectively) are presented. It can be seen that the algorithms derived using the approach proposed in this paper have considerably smaller complexity than the classical ones.

This work was motivated by the need of reduction of the number of summations in the cyclotomic fast Fourier transform algorithm [13]. Application of the method described here enabled considerable reduction of the number of operations needed by that algorithm.

IV. Conclusions

In this paper a method for construction of fast matrix-vector multiplication algorithms was proposed. The algorithms generated with this method have considerably smaller complexity compared to the classical algorithms. Open questions include development of an analytical estimate of their average complexity, as well as further improvement of the proposed method.

REFERENCES

[1] J. von zur Gathen and J. Gerhard, *Modern Computer Algebra*. Cambridge University Press, 1999.

[2] R. Blahut, *Fast Algorithms for Digital Signal Processing*. Addison-Wesley, 1985.

[3] I. Gohberg and V. Olshevsky, "Complexity of multiplication with vectors for structured matrices," *Linear Algebra and its Applications*, vol. 202, pp. 163–192, 1994. [Online]. Available: citeseer.ist.psu.edu/gohberg94complexity.html

[4] A. Aho, J. Hopcroft, and J. Ullman, *The design and analysis of computer algorithms*. Addison-Wesley, 1976.

[5] J. Cocke, "Global common subexpression elimination," *SIGPLAN Notices*, pp. 20–24, July 1970.

[6] W. E. Ryan, "An introduction to LDPC codes," in *CRC Handbook for Coding and Signal Processing for Recording Systems*, B. Vasic, Ed. CRC Press, 2004.

[7] M. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Efficient erasure correcting codes," *IEEE Transactions On Information Theory*, vol. 47, no. 2, February 2001.

[8] T. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Transactions On Information Theory*, vol. 47, no. 2, pp. 619–637, February 2001.

[9] R. Koetter and P. Vontobel, "Graph-covers and iterative decoding of finite-length codes," in *Proceedings of the 3rd International Conference on Turbo Codes and Related Topics*, 2003, pp. 75–82.

[10] M. Schwartz and A. Vardy, "On the stopping distance and the stopping redundancy of codes," *IEEE Transactions on Information Theory*, vol. 52, no. 3, pp. 922–932, March 2006.

[11] S. Sankaranarayanan and B. Vasic, "Iterative decoding of linear block codes: A parity-check orthogonalization approach," *IEEE Transactions on Information Theory*, vol. 51, no. 9, pp. 3347 – 3353, September 2005.

[12] A. Canteaut and F. Chabaud, "A new algorithm for finding minimum-weight words in a linear code: Application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511," *IEEE Transactions on Information Theory*, vol. 44, no. 1, pp. 367–378, January 1998.

[13] P. V. Trifonov and S. V. Fedorenko, "A method for fast computation of the Fourier transform over a finite field," *Problems of Information Transmission*, vol. 39, no. 3, pp. 231–238, July-September 2003, translation of Problemy Peredachi Informatsii.