Linear Algebra and its Applications 439 (2013) 1815–1824

Contents lists available at SciVerse ScienceDirect

Linear Algebra and its Applications

www.elsevier.com/locate/laa

Solving sparse linear systems of equations over finite fields using bit-flipping algorithm

Asieh A. Mofrad [a], M.-R. Sadeghi [a,*], D. Panario [b]

[a] Department of Mathematics and Computer Science, Amirkabir University of Technology, Tehran, Iran
[b] School of Mathematics and Statistics, Carleton University, K1S 5B6, Ontario, Canada

ARTICLE INFO

Article history:
Received 22 February 2012
Accepted 20 May 2013
Available online 7 June 2013
Submitted by E. Zerz

Keywords:
Bit-flipping
Sparse linear systems
Finite fields

ABSTRACT

Let $\mathbb{F}_q$ be the finite field with q elements. We give an algorithm for solving sparse linear systems of equations over $\mathbb{F}_q$ when the coefficient matrix of the system has a specific structure, here called *relatively connected*. This algorithm is based on a well-known decoding algorithm for low-density parity-check codes called bit-flipping algorithm. We modify and extend this hard decision decoding algorithm. The complexity of this algorithm is linear in terms of the number of columns n and the number of nonzero coefficients ω of the matrix per iteration. The maximum number of iterations is bounded above by m, the number of equations.

1. Introduction

Let $\mathbb{F}_q$ be the finite field with q elements where q is a prime power. In this paper we consider the classical problem of solving large sparse linear systems of equations of the form

$$A\mathbf{x} = \mathbf{b} \tag{1}$$

where $A = [a_{ij}]_{m\times n}$, $\mathbf{b} = [b_i]_{m\times 1}$ and $\mathbf{x} = [x_i]_{n\times 1}$ with a_{ij}, b_i, x_i in some algebraic structure for $1 \leqslant i \leqslant m$, $1 \leqslant j \leqslant n$. Our interest in this paper is when the elements belong to the finite field $\mathbb{F}_q$.

In general, classical Gaussian elimination method can be applied. For square matrices of dimension $n \times n$, Gaussian elimination takes time $O(n^3)$ and space $O(n^2)$ that makes it impractical for large and sparse systems. Over finite fields several algorithms have been proposed such as Wiedemann method [13] and its variants [4], the conjugate gradient and Lanczos algorithms (see [3,5,9]) and

* Corresponding author.
E-mail addresses: abolpour_asie@aut.ac.ir (A.A. Mofrad), msadeghi@aut.ac.ir (M.-R. Sadeghi), daniel@math.carleton.ca (D. Panario).

0024-3795/$ – see front matter
http://dx.doi.org/10.1016/j.laa.2013.05.016

also the structured Gaussian elimination method [9]. These algorithms can be used for linear systems modulo any integer g by applying the Chinese remainder theorem and Hensel lifting method [6]. These methods are probabilistic in nature and are designed for square matrices A. If the coefficient matrix of the system is an $m \times n$ matrix with $m \neq n$, the system has to be converted to a new system with a square coefficient matrix, so then the new system can be solved. In this paper we extend Gallager's bit-flipping (BF) algorithm [8] for solving sparse linear systems over finite field $\mathbb{F}_q$ when the coefficient matrix has a specific form that we call *relatively connected*. The bit-flipping is a low-complexity algorithm which was introduced for decoding LDPC (low-density parity-check) codes.

LDPC codes were first introduced by Gallager [8] in early 1960s and rediscovered by MacKay [11] in 1996. An LDPC code is a code such that its parity-check matrix H is sparse. In communications, when a vector $\mathbf{x}$ is sent through a channel, it is affected by noise (induced by the channel). Hence, a vector $\mathbf{y}$, which may be different from $\mathbf{x}$, is received. The decoding problem is, given a received vector $\mathbf{y}$, to find a good estimation for $\mathbf{x}$. The key points here are that $\mathbf{x}$ has to satisfy all parity-checks, that is, $H\mathbf{x}^{\mathbf{T}} = \mathbf{0}$ over $\mathbb{F}_q$ (in practice we are mostly interested in the case $q = 2$). When using LDPC codes an important issue is that the parity-check matrix H is sparse. When solving a sparse linear system, however, in contrast to the communications case, we do not have $\mathbf{y}$ as an input vector to the algorithm and so we start the algorithm with a random vector.

There are several LDPC algorithms based on *soft* and *hard* decisions [10]. In soft decision algorithms, we have input vectors that contain channel information in their components. This information is used when decisions are taken while executing the algorithm. Hard decision algorithms, in contrast, do not contain channel information in their components. Hence, since when solving systems of equations we do not deal with channels, we choose an effective hard decision algorithm like bit-flipping.

In [1], the bit-flipping algorithm is used for solving sparse linear systems of equations modulo a prime number p under the condition that the coefficient matrix has column degree at most 2. In the present paper we extend the algorithm to solve more general systems by introducing the *relatively connected* constraint. As a consequence, we have extended the work from matrices having column degree at most 2 to matrices having any column degree under the relatively connected constraint.

The structure of the paper is as follows. In Section 2, the general form of the bit-flipping decoding algorithm is explained. In Section 3, the extended bit-flipping algorithm is introduced. This algorithm can solve linear systems of equations over $\mathbb{F}_q$ that satisfy the relatively connected condition. This concept as well as examples are given in this section. The extended bit-flipping algorithm for systems over $\mathbb{F}_2$ is given in Section 4. This case is of practical importance, while at the same time easier to explain than the general case over $\mathbb{F}_q$ treated in Section 5. The cost of the algorithm is provided in Section 6. Finally, in Section 7, further directions of research are given.

2. The bit-flipping algorithm

For decoding binary LDPC codes we are given a received vector $\mathbf{y}$ and need to find a vector $\mathbf{x}$ such that $H\mathbf{x}^{\mathbf{T}} = \mathbf{0}$ over $\mathbb{F}_2$, or equivalently, $H\mathbf{x}^{\mathbf{T}} \equiv \mathbf{0} \pmod 2$. For this purpose, in the bit-flipping algorithm [8], the decoder first computes all the parity-checks; then, given a fixed parameter β called the *threshold*, the decoder changes any bit in the received vector $\mathbf{y}$ that is contained in more than β unsatisfied parity-check equations. The flipped bits are then used in the next iteration of the decoding process. The decoding algorithm stops when either all of the parity-checks are satisfied or a predefined maximum number of iterations is reached.

In contrast with choosing β as a fix number, adaptive threshold is suggested in [7]: if decoding fails for a given value of β, then the value of β can be reduced to allow further decoding iterations. Bit-flipping decoding with an adaptive threshold technique is given in detail in [2]. We use adaptive threshold technique but in a slightly different manner than in [2].

A simple bit-flipping decoding algorithm is given in Algorithm 1.

Algorithm 1 *(Bit-flipping algorithm).*
Input: parity-check equations and input vector $\mathbf{y}$.
Output: a solution vector.