A New Linear Programming Approach to Decoding Linear Block Codes

Kai Yang, *Student Member, IEEE*, Xiaodong Wang, *Senior Member, IEEE*, and Jon Feldman, *Member, IEEE*

***Abstract*—In this paper, we propose a new linear programming formulation for the decoding of general linear block codes. Different from the original formulation given by Feldman, the number of total variables to characterize a parity-check constraint in our formulation is less than twice the degree of the corresponding check node. The equivalence between our new formulation and the original formulation is proven. The new formulation facilitates to characterize the structure of linear block codes, and leads to new decoding algorithms. In particular, we show that any fundamental polytope is simply the intersection of a group of the so-called minimum polytopes, and this simplified formulation allows us to formulate the problem of calculating the minimum Hamming distance of any linear block code as a simple linear integer programming problem with much less auxiliary variables. We then propose a branch-and-bound method to compute a lower bound to the minimum distance of any linear code by solving a corresponding linear integer programming problem. In addition, we prove that, for the family of single parity-check (SPC) product codes, the fractional distance and the pseudodistance are both equal to the minimum distance. Finally, we propose an efficient algorithm for decoding SPC product codes with low complexity and maximum-likelihood (ML) decoding performance.**

***Index Terms*—Fractional distance, fundamental polytope, linear programming (LP) decoding, message-passing decoder, pseudodistance, single parity-check (SPC) product code.**

I. Introduction

RECENTLY, a linear programming (LP) approach has been proposed in [2], [3], and [1] for decoding binary linear codes. This method is based on an LP relaxation of the original maximum-likelihood (ML) decoding problem, which makes it simpler to analyze than the conventional message-passing decoders. For example, the concept of pseudocodeword arising from the LP decoding has been used to explain why iterative decoding algorithms for low-density parity-check (LDPC) codes such as the sum–product algorithm fail in certain scenarios [4]–[7]. Moreover, under the LP formulation, the decoder can be any algorithm that solves the LP exactly or approximately. For certain linear codes, one may exploit some specific structures to obtain efficient decoders. Given the large arsenal of powerful algorithms to solve very large-scale LP, the LP approach may lead to a paradigm shift in decoding error-correction codes. In fact, in [8], several LP decoders are developed for LDPC codes. In [9], an LP-based algorithm similar to the sum–product algorithm is proposed, and in [10], an adaptive LP decoding method is proposed to reduce the computational complexity of the LP decoder.

So far, most works in this area were based on the formulation given in [1]. In general, the number of auxiliary variables and constraints in this formulation increases exponentially with the maximum degree of the check nodes, which renders it inapplicable to linear codes with a high-density parity-check matrix.

The main contributions of this paper are threefold. First, we give a new LP formulation of decoding linear block code that is much simpler than the original formulation. In our formulation, the number of variables and constraints to characterize a parity-check constraint is less than twice the degree of the check node. We prove that this new formulation is equivalent to the original formulation. We introduce a new concept called the "*minimum polytope*." We show that the fundamental polytope of an arbitrary linear code is the intersection of a group of minimum polytopes. This formulation provides a means to characterize the structure of any linear block codes and offers insight into some fundamental problems such as how to calculate the minimum distance of any linear codes. Second, we show that, under the new LP formulation, the fractional distance of any linear block code can be easily calculated by solving a group of LPs. We also propose a branch-and-bound method to compute a nontrivial lower bound to the minimum distance of any linear block code. We further show that, for the family of SPC product codes, the fractional distance and pseudodistance are both equal to the minimum distance. Finally, based on the new formulation, we develop an efficient subgradient decoder that achieves the ML decoding performance for SPC product codes.

The remainder of this paper is organized as follows. In Section II, we present a new LP formulation of the decoding problem and prove the equivalence between the new formulation and the original one. In Section III, we discuss the calculation of the fractional distance and a lower bound to the minimum distance of any linear codes. In Section IV, we propose an efficient decoding algorithm for SPC product codes. Section V contains the conclusions.

II. New LP Formulation

Let $\mathbf{x}$ denote the transmitted codeword over the noisy channel, and $\mathbf{y}$ denote the received codeword. Given $\mathbf{y}$, the ML decoding problem is to obtain the estimated codeword that maximizes the likelihood of observing $\mathbf{y}$ under the channel model.

Manuscript received November 3, 2006; revised November 8, 2007. This work was supported in part by the U.S. National Science Foundation (NSF) under Grant DMS 0244583, and in part by the U.S. Office of Naval Research (ONR) under Grant N00014-03-1-0039.

K. Yang and X. Wang are with the Department of Electrical Engineering, Columbia University, New York, NY 10027 USA (e-mail: kyee@ee.columbia.edu; wangx@ee.columbia.edu).

J. Feldman was with the Department of Industrial Engineering and Operations Research, Columbia University, New York, NY 10027 USA. He is now with Google, New York, NY 10011 USA (e-mail: jonfeld@ieor.columbia.edu).

Communicated by I. Dumer, Associate Editor for Coding Theory.

Digital Object Identifier 10.1109/TIT.2007.915712

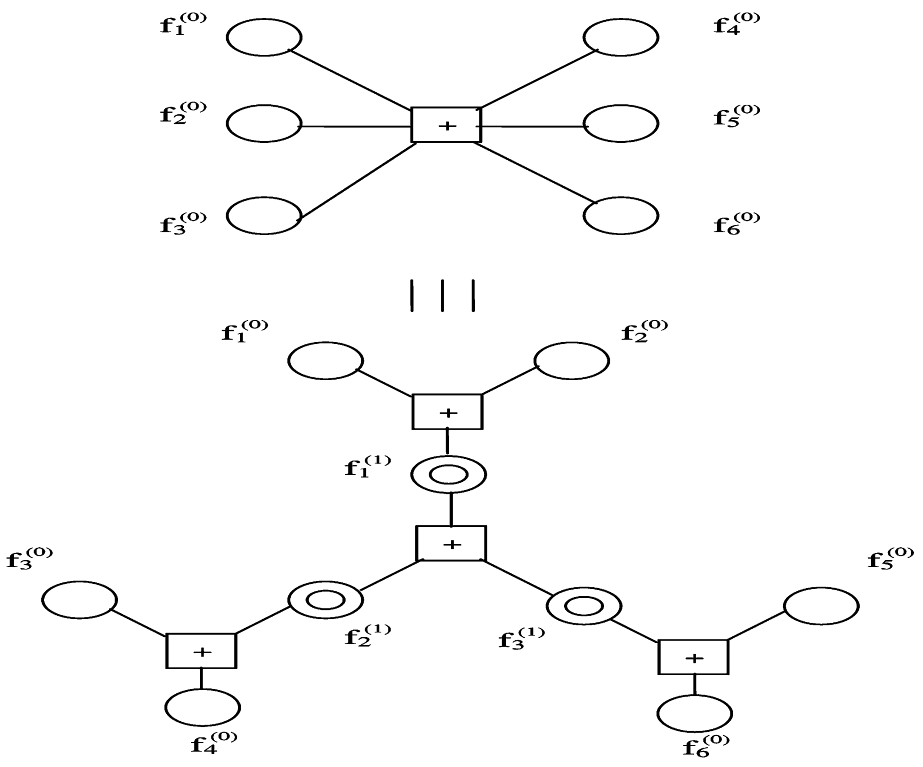

Fig. 1. Graph illustration of the proposed scheme to decompose a parity-check constraint with degree-6 into four parity-check constraints with degree-3.

Define ν_i as the log–likelihood ratio given by

$$\nu_i = \log \frac{\Pr[y_i|x_i = 0]}{\Pr[y_i|x_i = 1]}. \qquad (1)$$

The ML decoding over any binary memoryless symmetric channel corresponds to finding the codeword $\mathbf{f} = [f_1, f_2, \ldots, f_n]^T$ that minimizes $\sum_{i=1}^n \nu_i f_i$ [1]. We may interpret ν_i as the "cost" of decoding $f_i = 1$.

Denote the parity-check matrix of the binary linear code as $\mathbf{H} = [H_{ij}]_{m\times n}$; the integer programming formulation of the ML decoding problem is then given by

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} \nu_j f_j \\ \text{s.t.} \quad & \left[\sum_{j=1}^{n} H_{ij} f_j\right]_2 = 0 \qquad \forall\, i \quad \text{and} \quad f_j \in \{0,1\} \qquad \forall\, j \end{aligned} \qquad (2)$$

where $[\cdot]_2$ denotes the modular-2 operation.

A. Basic Idea

In this section, we present a cascaded integer programming formulation for the problem of ML decoding of a binary linear code. This leads directly to the LP formulation via relaxation.

The integer programming problem given in (2) is very hard to solve, because the polyhedron determined by the constraints cannot be described in polynomial time of the code length. A standard approach is to relax the original polyhedron to be some other convex set so that the transformed problem is tractable. However, both constraints in (2) are nonlinear, thus it is challenging to obtain a proper relaxation of the original problem.

In [1], the ML decoding problem is reformulated based on the corresponding Tanner graph. The key idea is to introduce auxiliary variables and to replace the constraints $[\sum_{j=1}^n H_{ij} f_j]_2 = 0$, $\forall\, i$, by certain linear equalities and inequalities. However, the number of constraints and variables in this formulation increases quickly with the degree of the check node.

Here, we propose a new formulation for the LP decoding problem. The number of variables and constraints in our new formulation is only about twice the degree of the check node. The basic idea is to decompose a high-degree check node into several low-degree check nodes. We can then use simple equalities and inequalities to represent the local constraints. This decomposition approach is demonstrated in Fig. 1. Considering the following parity-check constraint:

$$\left[f_1^{(0)}+f_2^{(0)}+f_3^{(0)}+f_4^{(0)}+f_5^{(0)}+f_6^{(0)}\right]_2 = 0, \qquad f_i^{(0)} \in \{0,1\} \qquad (3)$$

define auxiliary variables $f_1^{(1)}$, $f_2^{(1)}$, and $f_3^{(1)}$. It is easy to verify that the parity-check constraint in (3) is equivalent to the following set of constraints:

$$\begin{aligned} &\left[f_1^{(0)} + f_2^{(0)} + f_1^{(1)}\right]_2 = 0, \quad \left[f_3^{(0)} + f_4^{(0)} + f_2^{(1)}\right]_2 = 0, \\ &\left[f_5^{(0)} + f_6^{(0)} + f_3^{(1)}\right]_2 = 0, \quad \left[f_1^{(1)} + f_2^{(1)} + f_3^{(1)}\right]_2 = 0, \\ &\qquad\qquad f_i^{(0)}, f_i^{(1)} \in \{0,1\}. \end{aligned} \qquad (4)$$

The previous transformation is performed because each parity-check constraint $[f_1 + f_2 + f_3]_2 = 0$ can be characterized by a group of linear constraints which, together with the integer constraints $f_i \in \{0,1\}$, lead to an equivalent representation of the constraints in (3). Specifically, it can be verified that the parity-check constraints given in (4) are equivalent to the following group of constraints:

$$\begin{aligned}&f_1^{(0)}+f_2^{(0)}+f_1^{(1)} \le 2, \quad f_1^{(0)} \le f_2^{(0)}+f_1^{(1)},\\&f_2^{(0)} \le f_1^{(0)}+f_1^{(1)}, \quad f_1^{(1)} \le f_1^{(0)}+f_2^{(0)},\\&f_3^{(0)}+f_4^{(0)}+f_2^{(1)} \le 2, \quad f_3^{(0)} \le f_4^{(0)}+f_2^{(1)},\\&f_4^{(0)} \le f_3^{(0)}+f_2^{(1)}, \quad f_2^{(1)} \le f_3^{(0)}+f_4^{(0)},\\&f_5^{(0)}+f_6^{(0)}+f_3^{(1)} \le 2, \quad f_5^{(0)} \le f_6^{(0)}+f_3^{(1)}, \quad f_i^{(0)} \in \{0,1\}\\&f_6^{(0)} \le f_5^{(0)}+f_3^{(1)}, \quad f_3^{(1)} \le f_5^{(0)}+f_6^{(0)}, \quad f_i^{(1)} \in \{0,1\}.\end{aligned} \tag{5}$$

We can then relax the integer constraints $f_i^{(0)}, f_i^{(1)} \in \{0,1\}$ into the linear constraints $f_i^{(0)}, f_i^{(1)} \in [0,1]$ such that a new formation of the LP decoding problem can be obtained. A general statement of the previous transformation and the corresponding proofs are given next.

B. Cascaded Integer Programming Formulation of Ml Decoding

Assume there are γ_i ones in the ith row of the parity-check matrix $\mathbf{H}$ with the index $[\beta_1,\ldots,\beta_{\gamma_i}]$. Thus, the ith parity-check constraint can be written as

$$\left[\sum_{k=1}^{\gamma_i} f_{\beta_k}\right]_2 = 0, \qquad f_{\beta_k} \in \{0,1\}. \tag{6}$$

Let $f_k^{(0)} = f_{\beta_k}$, if γ_i is an even number, i.e., $\gamma_i = 2l^{(1)}$; similarly to the previous example, the original parity-check constraint in (6) can be rewritten as follows:

$$\begin{aligned}&\left[f_{2k-1}^{(0)}+f_{2k}^{(0)}+f_k^{(1)}\right]_2 = 0, \qquad k=1,2,\ldots l^{(1)}\\&\left[\sum_{k=1}^{l^{(1)}} f_k^{(1)}\right]_2 = 0, \qquad f_{2k-1}^{(0)}, f_{2k}^{(0)}, f_k^{(1)} \in \{0,1\}.\end{aligned} \tag{7}$$

If γ_i is an odd number, i.e., $\gamma_i = 2l^{(1)}-1$, we can have the similar transformation given by

$$\begin{aligned}&\left[f_{2k-1}^{(0)}+f_{2k}^{(0)}+f_k^{(1)}\right]_2 = 0, \qquad k=1,2,\ldots l^{(1)}-1\\&f_{2k-1}^{(0)} = f_k^{(1)}, \qquad k=l^{(1)}\\&\left[\sum_{k=1}^{l^{(1)}} f_k^{(1)}\right]_2 = 0, \qquad f_{2k-1}^{(0)}, f_{2k}^{(0)}, f_k^{(1)} \in \{0,1\}.\end{aligned} \tag{8}$$

Proposition 1: Assume there are γ_i ones in the ith row of the parity-check matrix $\mathbf{H}$, then let $[\beta_1,\ldots,\beta_{\gamma_i}]$ represent the index of 1s. Define $\Omega_0^{(i)} \triangleq \{\boldsymbol{f}=[f_1,\ldots,f_n] : 0 \le f_i \le 1$, $f_k^{(0)} = f_{\beta_k}, [f_1^{(0)},\ldots,f_{\gamma_i}^{(0)}]$ satisfy the constraints given in (6)$\}$, and $\Omega_1^{(i)} \triangleq \{\boldsymbol{f}=[f_1,\ldots,f_n] : 0 \le f_i \le 1, f_k^{(1)} = f_{\beta_k}$, $[f_1^{(1)},\ldots,f_{\gamma_i}^{(1)}]$ satisfy the constraints given in (7) or (8) $\}$. Then, $\Omega_0^{(i)} = \Omega_1^{(i)}$.

Proof: Assume γ_i is an even number, for any $\bar{\mathbf{f}} \in \Omega_0^{(i)}$, and let $f_k^{(0)} = f_{\beta_k}, \forall\, k$; it follows that $[\sum_{k=1}^{\gamma_i} f_k^{(0)}]_2 = 0$. We can generate $\{f_k^{(1)}\}$ by the following:

$$f_k^{(1)} = \left[f_{2k-1}^{(0)} + f_{2k}^{(0)}\right]_2, \qquad k=1,2,\ldots l^{(1)}. \tag{9}$$

Thus, we have $[\sum_{k=1}^{l^{(1)}} f_k^{(1)}]_2 = 0$, which means $\bar{\mathbf{f}} \in \Omega_1^{(i)}$. Therefore, we have $\Omega_0^{(i)} \subseteq \Omega_1^{(i)}$. Similarly, we can show that $\Omega_1^{(i)} \subseteq \Omega_0^{(i)}$. Thus, $\Omega_0^{(i)} = \Omega_1^{(i)}$. We can show that the proposition holds in a similar way for the case of an odd number. □

Moreover, if $l^{(1)} > 3$, we can recursively further decompose the parity-check constraint $\left[\sum_{k=1}^{l^{(1)}} f_k^{(1)}\right]_2 = 0$ similar to (7) or (8). Let $l^{(0)} = \gamma_i$. At the jth decomposition step, we have $l^{(j)} = \left\lceil \frac{l^{(j-1)}}{2} \right\rceil$. The transformation at the jth step can be written as

$$\begin{aligned}&\left[f_{2k-1}^{(j-1)}+f_{2k}^{(j-1)}+f_k^{(j)}\right]_2 = 0, \qquad k=1,2,\ldots,l^{(j)}-1\\&f_{2k-1}^{(j-1)} = f_k^{(j)}, \qquad k=l^{(j)},\\&\qquad\qquad \text{if } l^{(j-1)} = 2l^{(j)}-1\\&\left[f_{2k-1}^{(j-1)}+f_{2k}^{(j-1)}+f_k^{(j)}\right]_2 = 0, \qquad k=l^{(j)},\\&\qquad\qquad \text{if } l^{(j-1)} = 2l^{(j)}\\&\left[\sum_{k=1}^{l^{(j)}} f_k^{(j)}\right]_2 = 0\\&f_{2k-1}^{(j-1)}, f_{2k}^{(j-1)}, f_k^{(j)} \in \{0,1\}, \qquad j=1,\ldots,\kappa_i,\\&\qquad\qquad \text{where } \kappa_i = \left\lceil \log_2\left(\frac{\gamma_i}{3}\right)\right\rceil.\end{aligned} \tag{10}$$

Proposition 2: Assume there are γ_i 1s in the ith row of the parity-check matrix $\mathbf{H}$, and let $[\beta_1,\ldots,\beta_{\gamma_i}]$ represent the index of 1s. Denote $\Omega_j^{(i)} \triangleq \{\boldsymbol{f}=[f_1,\ldots,f_n] : 0 \le f_i \le 1, f_k^{(j)} = f_{\beta_k}, [f_1^{(j)},\ldots,f_{\gamma_i}^{(j)}]$ satisfy the constraints given in (10) $\}$. We have $\Omega_0^{(i)} = \Omega_1^{(i)} = \cdots = \Omega_{\kappa_i}^{(i)}$.

Proof: From Proposition 1, we have $\Omega_j^{(i)} = \Omega_{j+1}^{(i)}$, $0 \le j \le \kappa_i - 1$. Thus, $\Omega_0^{(i)} = \Omega_1^{(i)} = \cdots = \Omega_{\kappa_i}^{(i)}$. □

The previous decomposition approach can be treated as the extension of the reformulation of a Tanner graph in [11, Fig. 19]. However, the decomposition scheme here provides a more general hierarchical structure and rigorous mathematical proof. Let $\mathbf{v} = [\nu_1,\ldots,\nu_n]^T$ and $\mathbf{f} = [f_1,\ldots,f_n]^T$. Since $\mathbf{f}$ should satisfy all parity-check constraints, we have $\mathbf{f} \in \Omega_0^{(i)}, \forall\, i$. Thus, the original optimization problem given in (2) can be written as

$$\min\ \mathbf{v}^T\mathbf{f}, \qquad \text{s.t. } \mathbf{f} \in \Omega_0^{(1)} \cap \Omega_0^{(2)} \ldots \Omega_0^{(m)}. \tag{11}$$

From Proposition 2, we can replace any $\Omega_0^{(i)}$ by $\Omega_j^{(i)}, 1 \le j \le \kappa_i, \forall\, i$ without changing the solution. In particular, we can replace each $\Omega_0^{(i)}$ by $\Omega_{\kappa_i}^{(i)}$ and obtain the following optimization problem:

$$\min\ \mathbf{v}^T\mathbf{f}, \qquad \text{s.t. } \mathbf{f} \in \Omega_{\kappa_1}^{(1)} \cap \Omega_{\kappa_2}^{(2)} \ldots \Omega_{\kappa_m}^{(m)}. \tag{12}$$

C. New LP Formulation

The motivation of the LP decoding is to replace $\Omega_0^{(1)}, \Omega_0^{(2)}, \ldots \Omega_0^{(m)}$ in (11) by their corresponding convex hulls to obtain an LP problem, given by

$$\begin{aligned}&\min\ \mathbf{v}^T\mathbf{f}\\ &\text{s.t. } \mathbf{f} \in \operatorname{conv}\left(\Omega_0^{(1)}\right) \cap \operatorname{conv}\left(\Omega_0^{(2)}\right) \ldots \operatorname{conv}\left(\Omega_0^{(m)}\right). \quad (13)\end{aligned}$$

However, the set $\Omega_0^{(i)}$ is specified by the nonlinear parity-check constraint and integer constraints thus its convex hull cannot be obtained directly. One approach is to introduce auxiliary variables and replace the nonlinear parity-check constraint by a group of linear constraints so that the corresponding convex hull can be obtained by simply relaxing the integer constraints. It follows from Proposition 2 that $\Omega_0^{(i)} = \Omega_{\kappa_i}^{(i)}$, and therefore, $\operatorname{conv}(\Omega_0^{(i)}) = \operatorname{conv}(\Omega_{\kappa_i}^{(i)})$. We will show next that the convex hull of each set $\Omega_{\kappa_i}^{(i)}$ can be easily generated by simple inequality constraints, as in (5). We first give the following proposition.

Proposition 3: Denote $Q_1 \triangleq \{(z_1, z_2, z_3) : [z_1 + z_2 + z_3]_2 = 0, z_1, z_2, z_3 \in \{0, 1\}\}$, and $P_1 \triangleq \{(z_1, z_2, z_3) : z_1 \le z_2 + z_3, z_2 \le z_1 + z_3, z_3 \le z_1 + z_2, z_1 + z_2 + z_3 \le 2, z_1, z_2, z_3 \in \{0, 1\}\}$. Then, $Q_1 = P_1$ and the convex hull of P_1 is given by $C_1(z_1, z_2, z_3) \triangleq \{(z_1, z_2, z_3) : z_1 \le z_2 + z_3,\ z_2 \le z_1 + z_3,\ z_3 \le z_1 + z_2, z_1 + z_2 + z_3 \le 2, z_1, z_2, z_3 \in [0, 1]\}$. Similarly, denote $Q_2 \triangleq \{(z_1, z_2) : [z_1 + z_2]_2 = 0, z_1, z_2 \in \{0, 1\}\}$ and $P_2 \triangleq \{\{z_1, z_2\} : z_1 = z_2, z_1, z_2 \in \{0, 1\}\}$. Then, $Q_2 = P_2$ and the convex hull of P_2 is given by $C_2(z_1, z_2) \triangleq \{(z_1, z_2) : z_1 = z_2, z_1, z_2 \in [0, 1]\}$.

Proof: By simply enumerating all the feasible solutions, we have $Q_1 = P_1$ and $Q_2 = P_2$. In addition, the set of all extreme points of C_1 is equal to P_1, and the set of the extreme points of C_2 is equal to P_2. Thus, C_1 and C_2 are the convex hulls of P_1 and P_2, respectively. □

The previous proposition gives a compact representation of the convex hull of the set corresponding to the parity-check constraint with three elements. Specifically, the polytope C_1 is defined as *minimum polytope* hereafter. We will show in the following theorem that the convex hull of the set $\Omega_{\kappa_i}^{(i)}$ can be represented by the intersection of a group of minimum polytopes.

Theorem 4: Assume $\gamma_i \ge 3$. Let $\mathfrak{P}(\cdot)$ denote the projection onto the original space of $\boldsymbol{f}^{(0)}$. Denote $C_1^j(i)$ as the jth minimum polytope of the ith parity-check constraint, and $\tau(\gamma_i)$ as the number of minimum polytopes for the ith parity-check constraint. The convex hull of the set characterized by the ith parity-check constraint is equivalent to the intersection of a group of minimum polytopes projected onto the original space of $\boldsymbol{f}^{(0)}$, i.e.,

$$\Gamma_i \triangleq \operatorname{conv}\left(\Omega_0^{(i)}\right) = \operatorname{conv}\left(\Omega_{\kappa_i}^{(i)}\right) = \mathfrak{P}\left\{\bigcap_{j=1}^{\tau(\gamma_i)} C_1^j(i)\right\}. \quad (14)$$

Proof: It follows from Proposition 2 that $\Omega_0^{(i)} = \Omega_{\kappa_i}^{(i)}$, thus $\operatorname{conv}(\Omega_0^{(i)}) = \operatorname{conv}(\Omega_{\kappa_i}^{(i)})$. In addition, since the Tanner graph of $\Omega_{\kappa_i}^{(i)}$ is a tree, it follows from [5, Lemma 28] that the polytope of the LP relaxation problem is the convex hull of codewords, i.e., $\operatorname{conv}(\Omega_{\kappa_i}^{(i)}) = \mathfrak{P}\left\{\bigcap_{j=1}^{\tau(\gamma_i)} C_1^j(i)\right\}$. □

We assume $\gamma_i \ge 3$. Because the polytope C_2 for $\gamma_i = 2$ is simply $z_1 = z_2, 0 \le z_1 \le 1$, without loss of generality, we assume $\gamma_i \ge 3$ hereafter. In addition, from (10), it is seen $\left\lfloor \frac{\gamma_i}{2} \right\rfloor$ auxiliary variables are generated to reduce a parity-check with degree γ_i to a parity-check with degree $\left\lceil \frac{\gamma_i}{2} \right\rceil$. Denote the number of auxiliary variables to characterize a parity-check with degree γ_i as $\rho(\gamma_i)$. We show next $\rho(\gamma_i) = \gamma_i - 3$ and $\tau(\gamma_i) = \gamma_i - 2$. As shown in Fig. 1, the number of edges connected to the check node, the number of original variable nodes, and the number of auxiliary variable nodes are $3, 1, 2$, respectively. Thus, we have

$$3\tau(\gamma_i) = \gamma_i + 2\rho(\gamma_i). \quad (15)$$

In addition, the number of nodes in a tree is equal to the number of edges plus 1. Therefore, we have

$$1 + 3\tau(\gamma_i) = \gamma_i + \tau(\gamma_i) + \rho(\gamma_i). \quad (16)$$

From (15) and (16), we can obtain that $\rho(\gamma_i) = \gamma_i - 3$ and $\tau(\gamma_i) = \gamma_i - 2$.

Therefore, by relaxing the integer constraints in (12), we can obtain the following LP formulation:

$$\min\ \mathbf{v}^T\mathbf{f}, \qquad \text{s.t. } \mathbf{f} \in \Gamma_1 \cap \Gamma_2 \ldots \Gamma_m. \quad (17)$$

Because $\Gamma_j = \operatorname{conv}(\Omega_{\kappa_j}^{(j)}) = \operatorname{conv}(\Omega_0^{(j)})$, the optimization problem in (17) is equivalent to the original LP relaxation obtained in [1]. For simplicity, the original LP in [1] is denoted as LP and the new LP is denoted as C-LP (cascaded LP) hereafter. We assume every symbol of the codeword is involved in at least one parity check. Denote δ_l and δ_r, respectively, as the maximum degree of any variable node and check node in the Tanner graph. It has been shown that the number of variables and constraints in original LP formulation are $\mathcal{O}(n + m2^{\delta_r})$ and $\mathcal{O}\left(m + n\delta_l + 2n + 2m2^{\delta_r}\right)$, respectively [1]. A modified formulation has $\mathcal{O}\left(n + m\delta_r^2\right)$ variables and $\mathcal{O}(mn + n\delta_l\delta_r + 2n + 2m\delta_r^2)$ constraints [1]. On the other hand, because $\rho(\gamma_i) = \gamma_i - 3$, there are at most $(\delta_r - 3)$ auxiliary variables and δ_r original variables to characterize a parity-check constraint. Thus, at most $(2\delta_r - 3)$ variables are required to characterize a parity-check constraint. Moreover, we need $(\delta_r - 2)$ groups of constraints for each parity check, and there are four constraints in each group. Hence, for m parity-check constraints, the new C-LP formulation needs $\mathcal{O}(2m\delta_r - 3m)$ variables and $\mathcal{O}(4m\delta_r - 8m)$ constraints, which is of much lower complexity than original LP formulation.

Furthermore, as shown in [9], the idea of LP decoding is also to replace the local constraints enforced by each parity-check constraint with its corresponding convex hull. Thus, it leads to the same formulation as (17) and we have the following corollary.

Corollary 5: The C-LP is equivalent to the original LP, i.e., decoding using one formulation leads to the same result as using the other.

Therefore, the C-LP decoder shares the same properties of the LP decoder, such as the *ML-certificate* property and the *all-zero assumptions*[1]. In particular, it has been shown in [1, Th. 12] that the LP decoder is equivalent to the belief propagation (BP) decoder under the binary erasure channel (BEC). Hence, the following corollary follows immediately.

Corollary 6: Under the BEC, the C-LP decoder obtains the same result as given by the BP decoder.

Recall that n denotes the length of the codeword and γ_i denotes the degree of the ith parity-check constraint. Thus, for a parity-check matrix $\mathbf{H}$, we need $\gamma_v = \sum_{i=1}^{m}(\gamma_i - 3) + n$ variables and $\gamma_c = \sum_{i=1}^{m}(\gamma_i - 2)$ constraints to characterize it. Moreover, we can perform modular-2 summations between different rows of $\mathbf{H}$ to create more 0s and obtain a new parity-check matrix $\widehat{\mathbf{H}}$ without changing the codewords, which may further reduce the number of constraints and variables in the new formulation. However, note that such manipulation of changing the parity-check matrix may change the relaxed polytope of the LP decoder. After we submitted this paper, we found that Chertkov and Stepanov [12] proposed a similar approach independently and at about the same time as we did.

The idea of both C-LP and LP is to get a tight relaxation of each parity constraint, i.e., replacing $\Omega_0^{(i)}$ in the original constraints by its corresponding convex hull $\Gamma_i = \mathrm{conv}(\Omega_{\kappa_i}^{(i)}) = \mathrm{conv}(\Omega_0^{(i)})$, so that the local constraint of each check node is satisfied. Ideally, if we can get the convex hull of all the codeword, i.e., $\mathrm{conv}(\Omega_{\kappa_1}^{(1)} \cap \Omega_{\kappa_2}^{(2)} \ldots \Omega_{\kappa_m}^{(m)})$, then the ML codeword can be achieved by performing the LP decoding over this convex hull. It is known from [5, Lemma 28] that the LP relaxation is exact for trees. We can then obtain the following propositions.

Proposition 7: For a parity-check matrix $\boldsymbol{H}$, if it satisfies $\sum_{j=1}^{n} H_{i_1,j}H_{i_2,j} = 0$ for some i_1 and i_2, then we have $\mathrm{conv}(\Omega_{\kappa_{i_1}}^{(i_1)} \cap \Omega_{\kappa_{i_2}}^{(i_2)}) = \mathrm{conv}(\Omega_{\kappa_{i_1}}^{(i_1)}) \cap \mathrm{conv}(\Omega_{\kappa_{i_2}}^{(i_2)}) = \Gamma_{i_1} \cap \Gamma_{i_2}$.

Proposition 8: If there exists at most one "1" in each column of the parity-check matrix $\boldsymbol{H}$, we have $\mathrm{conv}(\bigcap_{i=1}^{m} \Omega_{\kappa_i}^{(i)}) = \bigcap_{i=1}^{m} \mathrm{conv}(\Omega_{\kappa_i}) = \bigcap_{i=1}^{m} \Gamma_i$. Thus, the pseudocodeword obtained by the LP decoder is the ML codeword.

However, in general, the polytope used in LP is not equivalent to the convex hull of the codewords, thus the ML decoding performance cannot be obtained. For codes with small length, we can employ efficient integer programming techniques such as the branch-and-bound method to improve the decoding performance [8].

III. Computing a Lower Bound of Minimum Distance

The minimum Hamming distance is a classical measure of code performance. However, the exact calculation of the minimum distance of an arbitrary code is very difficult. On the other hand, the fractional distance of an LDPC code can be efficiently calculated by solving the LP formulation given in [1], which servers as a lower bound of the minimum distance. However, the formulation in [1] cannot be used to compute the fractional distance of a code with high-density parity-check matrix due to its high complexity. In this section, we will show that our new formulation can be used to efficiently calculate the fractional distance of any linear block code at affordable complexity. In addition, although the fractional distance provides a nontrivial lower bound of the minimum distance, this bound may be loose for certain codes. Here, we develop a new method to calculate a better lower bound. This method is carried out in a multistage manner. At each stage, an improved lower bound over that of the previous stage is obtained by solving a group of LP problems. Then, we prove that for SPC product codes the fractional distance and pseudodistance are both equal to the minimum distance.

A. Calculation of Fractional Distance

Denote $\gamma_a = \sum_{i=1}^{m}(\gamma_i - 3)$ and $\gamma_c = \sum_{i=1}^{m}(\gamma_i - 2)$ as the total number of auxiliary variables and the number of minimum polytopes in the C-LP formulation. Denote $\mathbf{q} = \left[\mathbf{v}^T, \mathbf{0}_{1\times\gamma_a}\right]^T$ and $\mathbf{d} = \left[\mathbf{f}^T, \mathbf{u}_{1\times\gamma_a}\right]^T$, where $\mathbf{u}_{1\times\gamma_a}$ represents the auxiliary variables. Due to Propositions 2 and 3, the original integer programming problem given in (2) can be written as

$$\min\ \mathbf{q}^T\mathbf{d}, \qquad \text{s.t. } \mathbf{A}\mathbf{d} \preceq \mathbf{b};\quad d_i = \{0,1\} \qquad \forall\, i \tag{18}$$

where the matrix $\mathbf{A}$ is a $4\gamma_c \times (n+\gamma_a)$ matrix representing the first four constraints of each minimum polytope in Proposition 3 , and $\mathbf{b}$ is the corresponding right-hand side obtained from these constraints. $\mathbf{c} \preceq \mathbf{e}$ denotes elementwise inequalities, i.e., $c_i \le e_i$. Thus, the corresponding C-LP is given by

$$\min\ \mathbf{q}^T\mathbf{d}, \qquad \text{s.t. } \mathbf{A}\mathbf{d} \preceq \mathbf{b}, \qquad \mathbf{0} \preceq \mathbf{d} \preceq \mathbf{1}. \tag{19}$$

Any feasible solution of the LP decoding problem in (13) is defined as a pseudocodeword of the original decoding problem. Note here any codeword is a pseudocodeword. Moreover, the fractional distance is defined as the minimum weight of any nonzero pseudocodeword.

Let $\boldsymbol{w} \in \mathbb{R}_+^n$. The binary-input additive white Gaussian noise (BIAWGN) channel pseudodistance [5], [13] of $\boldsymbol{w}$ is defined to be

$$w_p^{\mathrm{BIAWGN}}(\boldsymbol{w}) \triangleq \frac{\|\boldsymbol{w}\|_1^2}{\|\boldsymbol{w}\|_2^2} \tag{20}$$

if $\boldsymbol{w} \ne \mathbf{0}$ and $w_p^{\mathrm{BIAWGN}}(\boldsymbol{w}) = 0$, otherwise, where $\|\boldsymbol{w}\|_1$ and $\|\boldsymbol{w}\|_2$ are the $\mathcal{L}_1$ and $\mathcal{L}_2$ norm of $\boldsymbol{w}$, respectively. Let $\boldsymbol{w}' \in \mathbb{R}_+^n$ be a vector with the same components as $\boldsymbol{w}$ but in nonincreasing order. Define

$$f(\xi) \triangleq \boldsymbol{w}_i', \qquad (i-1 < \xi \le i,\ 0 < \xi \le n)$$
$$F(\xi) \triangleq \int_0^{\xi} f(\xi')\mathrm{d}\xi', \quad e \triangleq F^{-1}\left(\frac{F(n)}{2}\right).$$

The BSC pseudodistance [5], [13] is defined to be $w_p^{\mathrm{BSC}}(\boldsymbol{w}) \triangleq 2e$ if $\boldsymbol{w} \ne \mathbf{o}$ and $w_p^{\mathrm{BSC}}(\boldsymbol{w}) \triangleq \mathbf{o}$, otherwise. The BEC pseudodistance [5], [13] is defined as the number of nonzero elements in $\boldsymbol{w}$.

In [1], the fractional distance of an LDPC code is calculated by solving a group of LP problems. An objective function is minimized over the facet of the original polytope. However, its complexity increases quickly with the degree of the check node. On the other hand, in our C-LP formulation, the facet of the

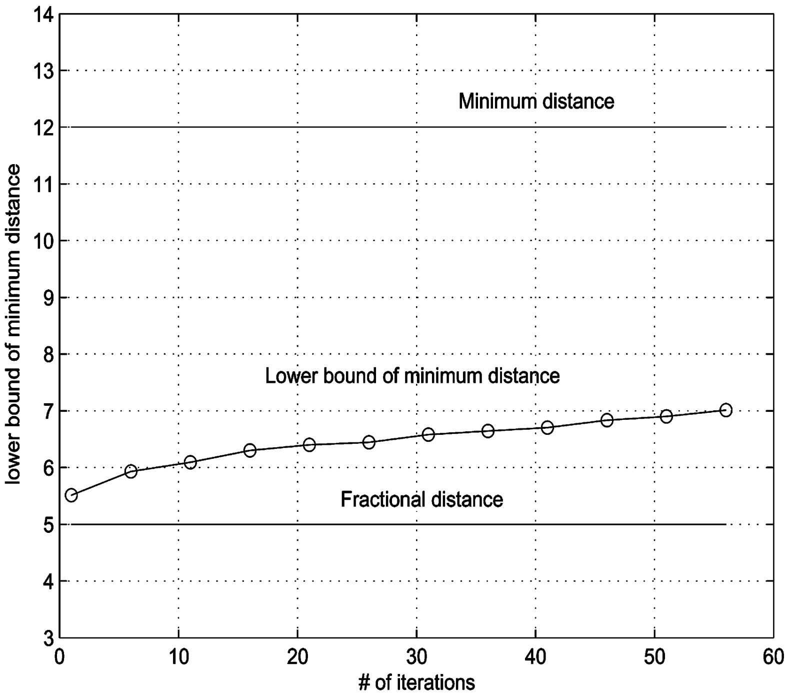

Fig. 2. Fractional distance, minimum distance, and the lower bound of the minimum distance obtained by a branch-and-bound method for a random LDPC code. Code length is 60, the rate is $\frac{1}{4}$, the left and right degrees of the corresponding Tanner graph are $\delta_l = 3$ and $\delta_r = 4$, respectively.

corresponding polytope can be easily obtained and the number of facets increases only linearly with the degree of check node. Thus, the fractional distance can be easily calculated. First, we give the following definition.

Definition 1: The facet of a polytope on which the all-zero codeword does not sit is defined as a nonzero facet.

The following proposition shows how to efficiently calculate the fractional distance of an arbitrary linear code.

Proposition 9: Any nonzero facet of the C-LP polytope must be one of the nonzero facets of its minimum polytopes.

Proof: Due to the definition of facets in [14], one of the inequalities in (19) must be active to form a facet of the C-LP polytope on which the zero codeword does not sit. Because the C-LP polytope is the intersection of a group of minimum polytopes, this inequality must be active for one of the facets of a minimum polytope on which the zero codeword does not sit. □

For a minimum polytope $C_1(z_1, z_2, z_3)$, the facet on which the zero codeword does not sit can only be formed by one of the following equalities, i.e., $z_1 = 1$, $z_2 = 1$, $z_3 = 1$, and $z_1 + z_2 + z_3 = 2$ together with C_1. In addition, since the first three equations are contained in the fourth equation as special cases, we only need to consider the facet formed by the cutting plan $z_1 + z_2 + z_3 = 2$. Therefore, we can easily compute the fractional distance of an arbitrary linear code by solving a group of LP problems. In addition, because the calculation of the minimum distance can be treated as a special case of the decoding problem in which all the log–likelihood ratios (LLRs) are set as 1, the adaptive branch-and-bound method proposed in [8] can be applied with minor modification. For example, we can select two bits $\{f_i, f_j\}$ out of the codeword and add one of the following three constraints into the original C-LP, i.e., $\{f_i = 1, f_j = 1\}$, $\{f_i = 0, f_j = 1\}$, and $\{f_i = 1, f_j = 0\}$. Denote the corresponding objective values as w_1, w_2, and w_3, respectively. We have $\ell_f \leq \min\{w_1, w_2, w_3\} \leq \ell_m$, where ℓ_f and ℓ_m denote the fractional distance and the minimum distance, respectively. Thus, we obtain a lower bound of the minimum distance. Notice this method is performed in a multistage manner and this lower bound can be improved by performing more stages, as given in [8], and will eventually converge to the Hamming distance.

The pseudodistance [5] is a better metric to characterize the performance of the LP decoding. Denote ℓ_f, ℓ_p, and ℓ_m as the fractional distance, pseudodistance, and minimum distance, respectively. In general, we have $\ell_f \leq \ell_p \leq \ell_m$, and thus the performance of the LP decoding is inferior to that of the ML decoding.

Numerical Example: The fractional distance, the minimum distance, and the lower bounds to the minimum distance of a random LDPC code are shown in Fig. 2. The length of the LDPC codeword is 60. The left degree is $\delta_l = 3$ and the right degree is $\delta_r = 4$. It is seen that the fractional distance is far less than the minimum distance, which may explain the performance gap between the ML and LP decoder for a random LDPC code as illustrated in [8]. On the other hand, it is seen that the branch-and-bound method can provide a nontrivial lower bound of the minimum distance. In addition, this lower bound can be improved successively as the number of iterations increases.

B. Distance Properties of SPC Product Codes

It has been observed in [8] that for the family of SPC product codes, the LP decoder offers the same performance as the ML decoder at high signal-to-noise ratio (SNR). Here, we show theoretically that for SPC product codes the fractional distance is equal to the minimum distance, i.e., $\ell_f = \ell_m$.

An SPC product code is a relatively large code built from smaller codeword blocks and its practical performance can be within 1 dB of the Shannon limit [15]. In addition, it has been shown that the asymptotic performance can be driven to zero within 2 dB of the capacity of the additive white Gaussian noise channel [16] for an SPC product code. A general SPC product code may have a different length in each dimension. Here, we consider only the (n_s, d_s) SPC product code [15], where n_s is the length of its component code in each dimension and d_s is its dimension. This code can be represented as a d_s-dimensional cubic and the component code in each dimension is an SPC code with block length n_s. Thus, the length of this code is $n_s^{d_s}$ and its rate is $\left(\frac{n_s-1}{n_s}\right)^{d_s}$. For example, the 2-D SPC product code consists of a data block, parity checks on the rows, parity checks on the columns, and checks on checks. Note that although the rate of this code approaches zero when the block length grows to infinity, it can achieve near-capacity performance for practical block length and code rates [15]. In addition, asymptotic analysis has shown that there exist SPC product codes with nonzero rate which can approach the channel capacity when the block length goes to infinity [16]. In order to show $\ell_f = \ell_m$, the following two corollaries are needed.

Corollary 10: The fractional distance of the fundamental polytope of any SPC code is equal to 2.

Proof: The parity-check matrix of any SPC code contains exactly one "1" in each column. Thus, it follows from Proposition 8 that the pseudocodeword is always the codeword of the SPC code. Therefore, its fractional distance is equal to 2, which is the minimum distance of any SPC code. □

Corollary 11: Any feasible solution $\boldsymbol{f}$ to the LP decoding problem given in (17) for any SPC code is either an all-zero codeword or it contains at least two nonzero variables, i.e., $f_i > 0$ and $f_j > 0, 1 \leq i, j \leq n$.

Proof: Because the polytope given in (17) of any SPC code is a convex hull, any point inside can be expressed as a linear combination of its extreme points. It follows from Corollary 10 that any nonzero extreme point contains at least two nonzero elements. Therefore, if $\mathbf{f}$ is not an all-zero codeword, it must contain at least two nonzero elements. □

Based on the previous two corollaries, we are ready to prove the following theorem.

Theorem 12: The fractional distance of the fundamental polytope of any d_s-dimensional SPC product code is equal to its minimum distance, i.e., $\ell_f = \ell_m = 2^{d_s}$.

Proof: It follows from Corollary 11 that Theorem 12 holds for $d_s = 1$. We will prove this theorem by induction. Suppose that it is true for $d_s = \eta$. It follows from Corollary 11 that each nonzero pseudocodeword of the $(n_s, \eta + 1)$ SPC product code contains at least two nonzero variables. Due to the special structure of the SPC product code, an $(n_s, \eta + 1)$ SPC product code can be decomposed into n_s component codes where each component code is an (n_s, η) SPC product code. This decomposition can be performed along each of the $\eta + 1$ dimensions and we can always find a decomposition scheme for which the two nonzero variables fall into different component codes. Assume the fundamental polytopes of the original polytope and the two component codes are Q_o, Q_a, and Q_b, respectively. In addition, let ϕ_o, ϕ_a, and ϕ_b denote the set containing all possible cutting plans $\varphi \triangleq \{z_1, z_2, z_3 : z_1 + z_2 + z_3 = 2\}$ for polytopes Q_o, Q_a, and Q_b, respectively. The codewords of the $(n_s, \eta+1)$ SPC product code and two component codes are denoted as $\mathbf{f}_o = [f_{o,1}, f_{o,2}, \ldots, f_{o,n_s^{\eta+1}}]^T$, $\mathbf{f}_a = [f_{a,1}, f_{a,2}, \ldots, f_{a,n_s^{\eta}}]^T$, and $\mathbf{f}_b = [f_{b,1}, f_{b,2}, \ldots, f_{b,n_s^{\eta}}]^T$, respectively. From the definition of fractional distance, we have

$$\begin{aligned}
&\ell_f(n_s, \eta + 1) \\
&= \min_{\varphi \in \phi_o} \min_{\mathbf{f}_o \in Q_o \cap \varphi} \sum_i f_{o,i} && (21)\\
&\geq \min_{\varphi \in \phi_o} \min_{\mathbf{f}_o \in Q_o \cap \varphi,\ \mathbf{f}_a \neq \mathbf{0},\ \mathbf{f}_b \neq \mathbf{0}} \left(\sum_i f_{a,i} + \sum_j f_{b,j} \right) && (22)\\
&= \min_{\varphi \in \phi_a} \min_{\mathbf{f}_a \in Q_a \cap \varphi} \sum_i f_{a,i} + \min_{\varphi \in \phi_b} \min_{\mathbf{f}_b \in Q_b \cap \varphi} \sum_j f_{b,j} && (23)\\
&= 2^\eta + 2^\eta = 2^{\eta+1}. && (24)
\end{aligned}$$

Here, the first equation is derived from the definition of fractional distance and Proposition 9. Equation (22) is obtained because of $f_i \geq 0$, and Corollary 11, i.e., any nonzero pseudocodeword contains at least two nonzero variables. Because $\phi_a, \phi_b \subset \phi_o$, $Q_a, Q_b \subset Q_o$, $\mathbf{f}_a \neq \mathbf{0}$, and $\mathbf{f}_b \neq \mathbf{0}$, we have (23). Equation (24) is derived by the assumption that the fractional distance of a (d_s, η) SPC product code is 2^η. Thus, the fractional distance of an $(n_s, \eta + 1)$ code is at least twice of the fractional distance of an (n_s, η) code, i.e., $2^{\eta+1}$. It is known that the minimum weight of an $(n_s, \eta + 1)$ SPC product code is $2^{\eta+1}$ [15]. Therefore, the fractional distance of any d_s-dimensional SPC product code is equal to 2^{d_s}. □

Because the BIAWGN, BSC, and BEC pseudodistances of any linear block code are lower bounded by the corresponding fractional distances, we then have the following corollary.

Corollary 13: The BIAWGN, BSC, and BEC pseudodistances of any SPC product codes are equal to 2^{d_s}.

The concept of pseudoweight spectrum gap has been introduced in [17] to characterize the performance gap of the ML decoder and LP decoder. The pseudoweight spectrum gap can be loosely defined as the difference between the smallest pseudoweight of the noncodeword pseudocodeword and the minimum distance. If the pseudoweight spectrum gap is strictly positive, the performance of the LP decoder will converge to that of the ML decoder when SNR grows to infinity. Since the fractional distance of the class of codes under consideration is equal to their minimum distance, it is easy to obtain the following proposition.

Proposition 14: If the fractional distance of the fundamental polytope of a code is equal to its Hamming distance, the BIAWGN pseudoweight spectrum gap of this code is strictly positive.

Proof: Recall that ℓ_m is the minimum Hamming distance. For any noncodeword pseudocodeword $\boldsymbol{w} = [w_1, w_2, \ldots, w_n]^T$, it must contain at least one noninteger variable. Because $0 \le w_i \le 1, \forall\, i$, we have $||\boldsymbol{w}||_2^2 < ||\boldsymbol{w}||_1$. Therefore, $w_p^{\text{BIAWGN}}(\boldsymbol{w}) \triangleq \frac{||\boldsymbol{w}||_1^2}{||\boldsymbol{w}||_2^2} > ||\boldsymbol{w}||_1 = \ell_m$. Thus, the pseudoweight spectrum gap is strictly bigger than zero. □

Because the fractional distance of the fundamental polytope of any SPC product is equal to its Hamming distance, we have the following corollary.

Corollary 15: The BIAWGN pseudoweight spectrum gap of the SPC product code is strictly positive.

IV. Fast Decoder for SPC Product Codes

Typically, general LP problems are solved by the interior-point method or the simplex method [18] and their variations. On the other hand, some codes exhibit specific structures that can be exploited to devise more efficient decoding algorithms. For example, low-complexity algorithms based on the LP formulation are proposed in [8]–[10] for LDPC codes.

In this section, we develop a fast decoding method for the SPC product code based on the Lagrange dual method for integer programming problem (cf. [18, Ch. 11]). Such technique provides a better approximation to the original ML decoding problem than the LP decoder. In particular, for the class of SPC product code under consideration, this dual approach will always converge to an ML codeword if the LP decoder yields an integral solution. If the LP decoder reaches a fractional solution, the dual approach will converge to a solution with same objective function of this fractional solution (cf. [18, Th. 11.4, Ch. 11]). Note that the parity-check matrix of an (n_s, d_s) SPC product code can be decomposed into d_s submatrices such that each submatrix satisfies the conditions for Proposition 8; that is, there exists at most 1 in each column of the matrix. Hence, $\mathbf{A}$ and $\mathbf{b}$ in (18) and (19) can be decomposed as $\mathbf{A} = \left[\mathbf{A}_1^T, \ldots, \mathbf{A}_{d_s}^T\right]^T$ and $\mathbf{b} = \left[\mathbf{b}_1^T, \ldots, \mathbf{b}_{d_s}^T\right]^T$, and the optimization problem in (19) can be reformulated as

$$\min \mathbf{q}^T\mathbf{d}, \qquad \text{s.t.}\, \mathbf{A}_1\mathbf{d} \preceq \mathbf{b}_1; \ldots; \mathbf{A}_{d_s}\mathbf{d} \preceq \mathbf{b}_{d_s}; \mathbf{0} \preceq \mathbf{d} \preceq \mathbf{1}. \tag{25}$$

By employing the Lagrange dual method [19], for a given vector $\mathbf{p} \succeq \mathbf{0}$, (25) can be transformed into the following Lagrange problem:

$$L_1(\mathbf{p}) = \min_{\mathbf{A}_1\mathbf{d} \preceq \mathbf{b}_1;\, \mathbf{0} \preceq \mathbf{d} \preceq \mathbf{1}} \mathbf{q}^T\mathbf{d} - \mathbf{p}^T(\bar{\mathbf{b}}_1 - \bar{\mathbf{A}}_1\mathbf{d}) \tag{26}$$

where $\bar{\mathbf{A}}_1$ consists of all submatrices of $\mathbf{A}$ other than $\mathbf{A}_1$, i.e., $\bar{\mathbf{A}}_1 = \mathbf{A}/\mathbf{A}_1$ and $\bar{\mathbf{b}}_1 = \mathbf{b}/\mathbf{b}_1$. In addition, denote $\mathbf{H}_1$ as the parity-check submatrix corresponding to $\mathbf{A}_1$. Define the auxiliary variables $\mathbf{u} = \left[\mathbf{u}_1^T, \bar{\mathbf{u}}_1{}^T\right]^T$, where $\mathbf{u}_1$ corresponds to the auxiliary variables generated by the parity-check matrix $\mathbf{H}_1$. Recall that $\mathbf{d} = [\mathbf{f}^T, \mathbf{u}^T]^T$. Because the submatrix of $\bar{\mathbf{A}}_1$ corresponding to $\mathbf{u}_1$ is a zero matrix, we have $\bar{\mathbf{A}}_1\mathbf{d} = \hat{\mathbf{A}}_1\left[\mathbf{f}^T, \bar{\mathbf{u}}_1^T\right]^T$, where $\hat{\mathbf{A}}$ is the submatrix of $\mathbf{A}$ corresponding to $\left[\mathbf{f}^T, \bar{\mathbf{u}}_1^T\right]^T$. Similarly, the submatrix of $\mathbf{A}_1$ corresponding to $\bar{\mathbf{u}}_1$ is a zero matrix, therefore, we have $\mathbf{A}_1\mathbf{d} = \tilde{\mathbf{A}}_1\left[\mathbf{f}^T, \mathbf{u}_1^T\right]^T$, where $\tilde{\mathbf{A}}$ is the submatrix of $\mathbf{A}_1$ corresponding to $\left[\mathbf{f}^T, \mathbf{u}_1^T\right]^T$. On the other hand, due to Proposition 8, the pseudocodeword of the LP decoding problem given in (26) must be integer; and thus, the constraints $\mathbf{A}_1 d \preceq \mathbf{b}_1, \mathbf{0} \preceq \mathbf{d} \preceq \mathbf{1}$, are equivalent to the following constraints:

$$\begin{aligned}\left[\mathbf{H}_1^T\mathbf{f}\right]_2 &= \mathbf{0}, \qquad f_i \in \{0,1\} \qquad \forall\, i\\ \tilde{\mathbf{A}}_1\left[\mathbf{f}^T, \mathbf{u}_1^T\right]^T &\preceq \mathbf{b}_1, \qquad \mathbf{0} \preceq \mathbf{u}_1 \preceq \mathbf{1}.\end{aligned} \tag{27}$$

Let $\mathbf{p}^T\bar{\mathbf{A}}_1\mathbf{d} = \mathbf{p}^T\hat{\mathbf{A}}_1\left[\mathbf{f}^T, \bar{\mathbf{u}}_1^T\right]^T = \mathbf{a}^T\mathbf{f} + \bar{\mathbf{a}}^T\bar{\mathbf{u}}_1$, where $\hat{\mathbf{A}}_1^T\mathbf{p} = [\mathbf{a}^T, \bar{\mathbf{a}}^T]^T$. Denote $\bar{\mathbf{v}} = \mathbf{v} + \mathbf{a}$, because $\mathbf{q}^T\mathbf{d} = \mathbf{v}^T\mathbf{f}$. Equation (26) can be reformulated as

$$L_1(\mathbf{p}) = -\mathbf{p}^T\bar{\mathbf{b}}_1 + E + F. \tag{28}$$

The second optimization problem in (28) is given by

$$F \triangleq \min_{\mathbf{0} \preceq \bar{\mathbf{u}}_1 \preceq \mathbf{1}} \bar{\mathbf{a}}^T\bar{\mathbf{u}}_1 \tag{29}$$

which can be easily solved by setting the $\bar{u}_1(i) = 0$ if $\bar{a}(i) > 0$ and $\bar{u}_1(i) = 1$ if $\bar{a}(i) < 0$. For the first optimization problem in (28)

$$E \triangleq \min_{\left[\mathbf{H}_1^T\mathbf{f}\right]_2 = \mathbf{0},\, f_i \in \{0,1\}, \forall\, i;\, \tilde{\mathbf{A}}_1\left[\mathbf{f}^T, \mathbf{u}_1^T\right]^T \preceq \mathbf{b}_1,\, \mathbf{0} \preceq \mathbf{u}_1 \preceq \mathbf{1}} \bar{\mathbf{v}}^T\mathbf{f}. \tag{30}$$

Because the vectors $\bar{\boldsymbol{v}}$ and $\boldsymbol{f}$ are independent of $\bar{\boldsymbol{u}}_1$, we have

$$\min_{\left[\mathbf{H}_1^T\mathbf{f}\right]_2 = \mathbf{0},\, f_i \in \{0,1\}, \forall\, i;\, \tilde{\mathbf{A}}_1\left[\mathbf{f}^T, \mathbf{u}_1^T\right]^T \preceq \mathbf{b}_1,\, \mathbf{0} \preceq \mathbf{u}_1 \preceq \mathbf{1}} \bar{\mathbf{v}}^T\mathbf{f} = \min_{\left[\mathbf{H}_1^T\mathbf{f}\right]_2 = \mathbf{0},\, f_i \in \{0,1\}, \forall\, i} \bar{\mathbf{v}}^T\mathbf{f} \tag{31}$$

and $\mathbf{u}_1$ is determined by

$$\tilde{\mathbf{A}}_1\left[\mathbf{f}^T, \mathbf{u}_1^T\right]^T \preceq \mathbf{b}_1, \qquad \mathbf{0} \preceq \mathbf{u}_1 \preceq \mathbf{1}. \tag{32}$$

Because $\mathbf{H}_1$ satisfies Proposition 8, every parity-check of this submatrix is independent of the others. Thus, the optimization problem in (31) can be fully decomposed into independent subproblems as follows. Suppose that there are γ_i ones in the ith row of $\mathbf{H}_1$, located at $\{\beta_1, \ldots, \beta_{\gamma_i}\}$. The subproblem can be written as

$$\min \sum_{k=1}^{\gamma_i} \bar{\nu}_{\beta_k} f_{\beta_k}, \qquad \text{s.t.} \left[\sum_{k=1}^{\gamma_i} f_{\beta_k}\right]_2 = 0, \qquad f_{\beta_k} \in \{0,1\}. \tag{33}$$

This subproblem can be efficiently solved by the following Wagner subroutine [20].

```
Procedure Parity-Decoding ($\{\bar{\nu}_{\beta_k}\}_{k=1}^{\gamma_i}$).
  • Set $\widehat{f}_{\beta_k} = \frac{-\text{sign}(\bar{\nu}_{\beta_k}) + 1}{2}$.
      — If $\left[\sum_{k=1}^{\gamma_i} \widehat{f}_{\beta_k}\right]_2 = 0$, stop.
      — Otherwise, let $\widehat{f}_{\beta_j} \leftarrow 1 - \widehat{f}_{\beta_j}$, where $\beta_j$ is one of
        the elements of the set $\text{argmin}(\text{abs}(\bar{\nu}_{\beta_k}))$.
  • Output $\{\widehat{f}_{\beta_k}\}_{k=1}^{\gamma_i}$.
```

Assume there are $\bar{m}$ rows in the submatrix $\mathbf{H}_1$. We can simply run the previous subroutine in parallel for each row to solve the first optimization problem in (28). After we generate the codeword $\mathbf{f}$, we can get the auxiliary variables $\mathbf{u} = \left[\mathbf{u}_1^T, \bar{\mathbf{u}}_1{}^T\right]^T$ from the solution to (32) and (29).

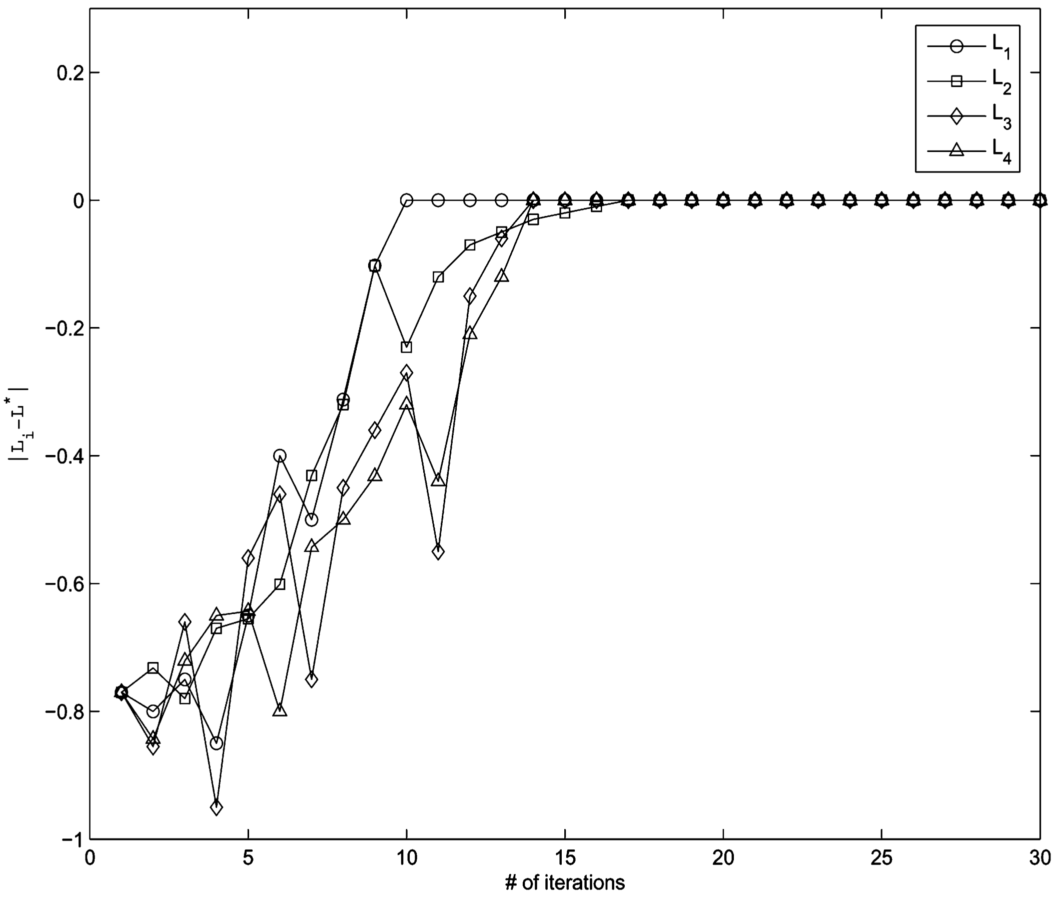

Fig. 3. Convergence behavior of the subgradient decoders against the iteration number for a $(4, 4)$ SPC product code. L^* denotes the optimal objective value of the problem given in (35). L_j, $1 \le j \le 4$, denotes the objective values against the iteration number for the jth subgradient decoder.

Assume that the optimal solution to the original C-LP is $\mathbf{d}^*$. We have $L_1(\mathbf{p}) \le \mathbf{q}^T\mathbf{d}^*$. In order to obtain the optimal solution to the original C-LP problem, we need to solve the following problem:

$$\max\ L_1(\mathbf{p}), \quad \text{s.t. } \mathbf{p} \succeq \mathbf{0}. \tag{34}$$

This problem can be solved by the subgradient method. A subgradient of a concave function $L_1(\cdot)$ at $\mathbf{p}$ is a vector $\mathbf{h}$ such that $L_1(\mathbf{g}) \le L_1(\mathbf{p}) + \mathbf{h}^T(\mathbf{g} - \mathbf{p})$ for all $\mathbf{g}$. Denote $\tilde{\mathbf{d}}$ as the solution of (28), a subgradient of the function $L_1(\mathbf{p})$ is given by $\bar{\mathbf{b}}_1 - \bar{\mathbf{A}}_1\tilde{\mathbf{d}}$. Therefore, we can obtain the following subgradient decoding algorithm.

Algorithm (Dual decomposition method for C-LP decoding)

```
1) Set $t = 0$ and $\boldsymbol{p}^0 = \mathbf{0}$.
2) Solve the primal problem given in (28) as outlined previously to obtain $\boldsymbol{d}^t$.
3) Calculate the subgradient $\boldsymbol{s}^t = \boldsymbol{b}_2 - \boldsymbol{A}_2\boldsymbol{d}^t$.
4) If $\|\boldsymbol{s}^t\| \le \epsilon$, stop; otherwise, let $\boldsymbol{p}^{t+1} = [\boldsymbol{p}^t + \theta_t\boldsymbol{s}^t]_+$, $t = t + 1$; go to step 2).
```

Here, $[\cdot]_+$ denotes the projection onto the nonnegative orthant and ϵ is a predefined threshold. θ_t is a stepsize; and if it satisfies certain conditions such as those given in [18, p. 505], then this method converges.

Furthermore, we may employ a joint subgradient decoder to improve the decoding performance. In this decoder, we can select one group of constraints in (25) as the hard constraints and move the other constraints to the objective function, leading to the following problem that is similar to (26):

$$L_j(\mathbf{p}) = \min_{\mathbf{A}_j\mathbf{d} \preceq \mathbf{b}_j;\ \mathbf{0} \preceq \mathbf{d} \preceq \mathbf{1}} \mathbf{q}^T\mathbf{d} + \mathbf{p}^T(\bar{\mathbf{b}}_j - \bar{\mathbf{A}}_j\mathbf{d}). \tag{35}$$

Note that the solution of this method is integral, because the Wagner decoder always outputs an integral solution. If this integral solution is a feasible codeword, it must be the optimal solution of the ML decoding problem. However, in practice, it may not converge to a feasible codeword because we have moved part of the constraint to the objective function. In addition, we can run a given number of iterations of the previous subgradient decoding algorithm for each problem in (35) and then the one with the maximum objective value is taken as the final result. Such a method can provide a diversity gain and thereby improve the decoding performance. Note that the proposed subgradient decoder is based on a dual-decomposition approach, which is different from the subgradient methods used in [1] and [9].

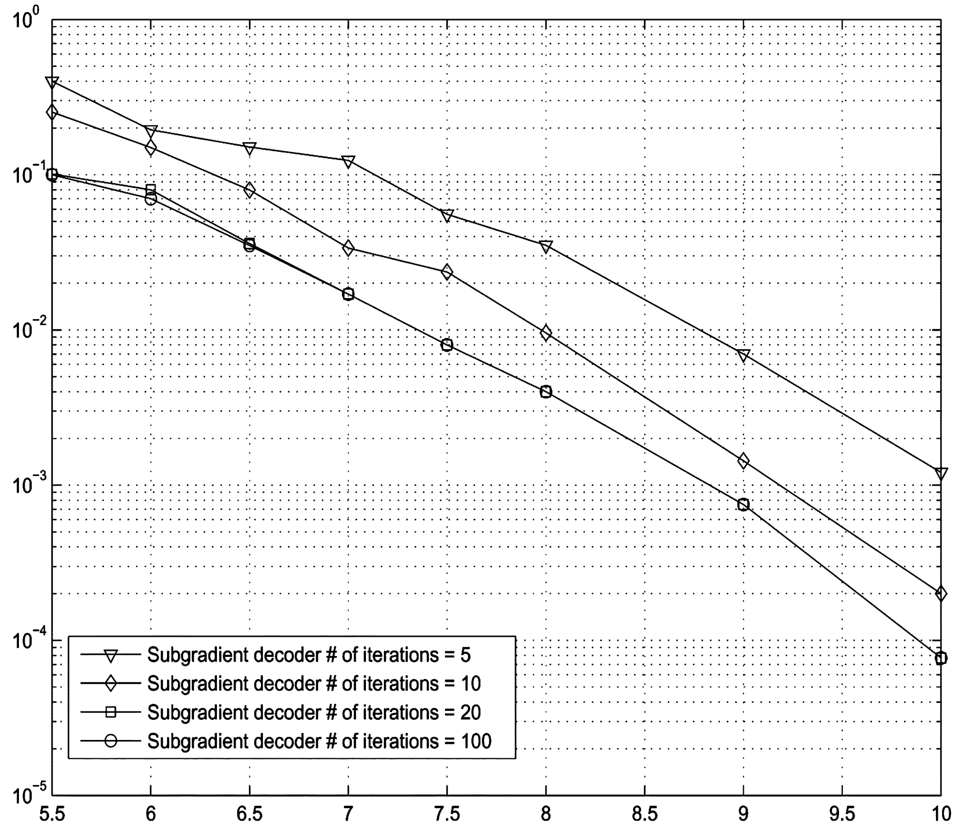

Fig. 4. WER performance of the subgradient decoder with different number of iterations. A $(5, 2)$ SPC product code is used. The code length is 25 and rate is $\left(\frac{4}{5}\right)^2 = 0.64$. The average WER is obtained by counting 100 decoding errors in the simulations.

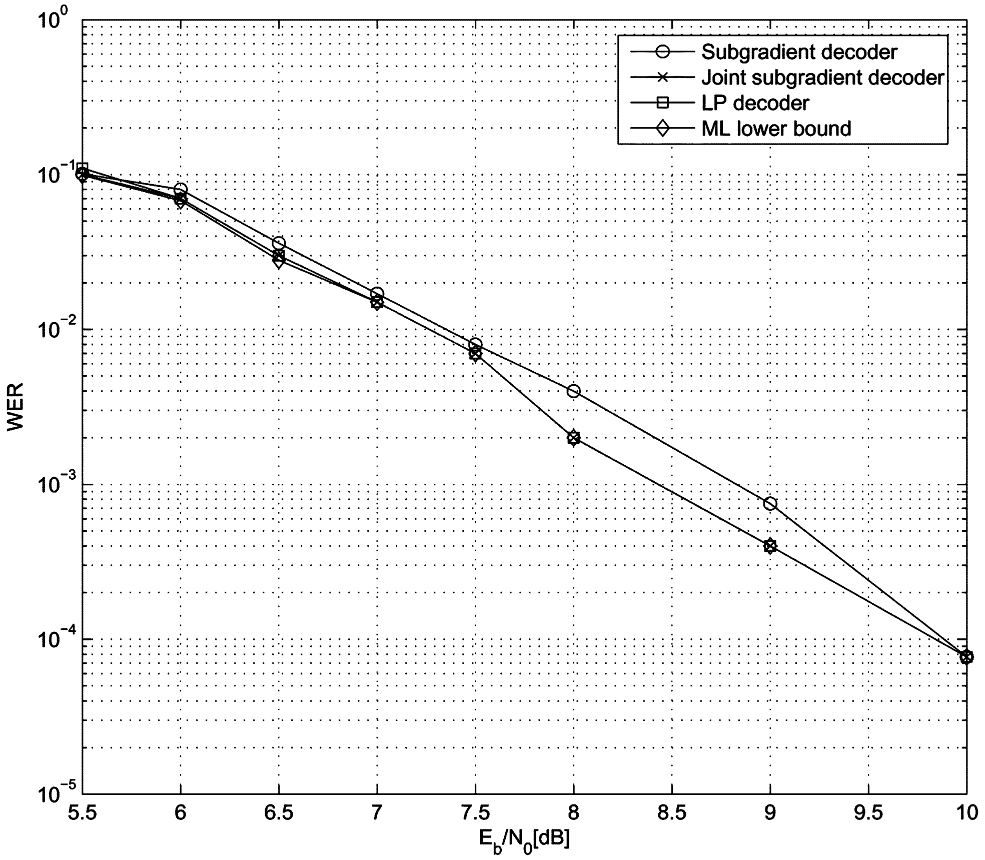

Fig. 5. WER performance of the subgradient decoder, joint subgradient decoder, the LP decoder, and the corresponding ML lower bound. The number of iterations for subgradient decoders is set as 100. A $(5, 2)$ SPC product code is used. The code length is 25 and rate is $\left(\frac{4}{5}\right)^2 = 0.64$. The average WER is obtained by counting 100 decoding errors in the simulations.

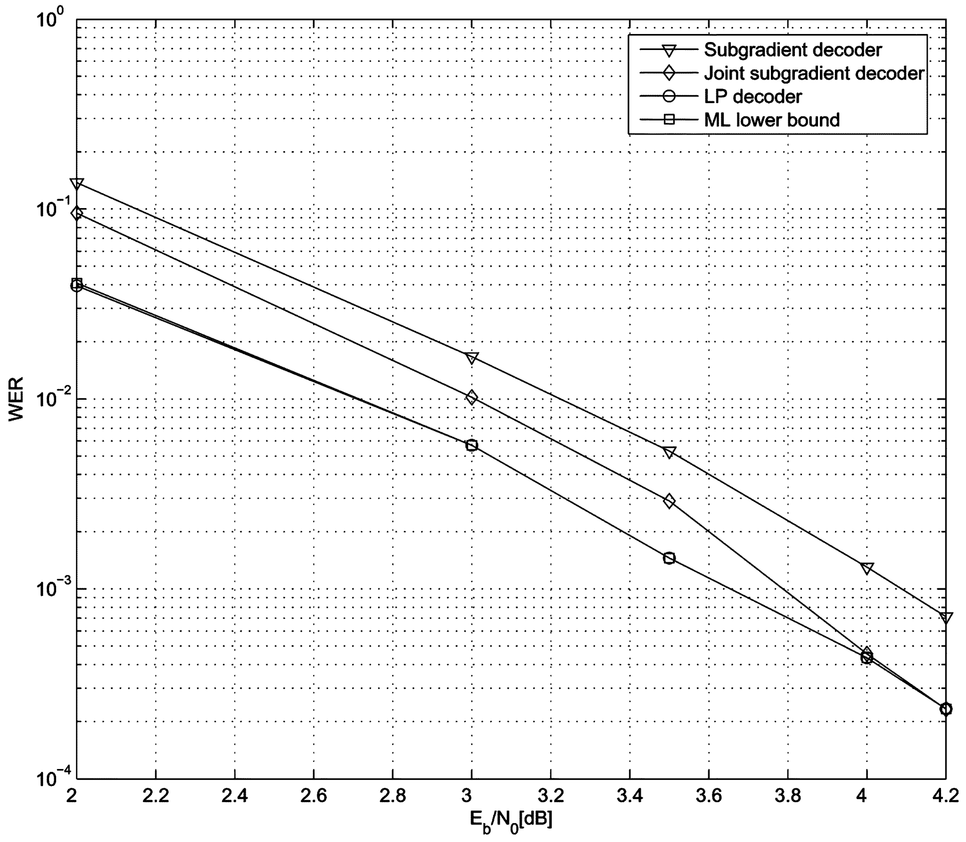

Fig. 6. WER performance of the subgradient decoder, joint subgradient decoder, the LP decoder, and the corresponding ML lower bound. A $(4, 4)$ SPC product code is used. The code length is 256 and rate is $\left(\frac{3}{4}\right)^4 \approx 0.32$. The average WER is obtained by counting 100 decoding errors in the simulations.

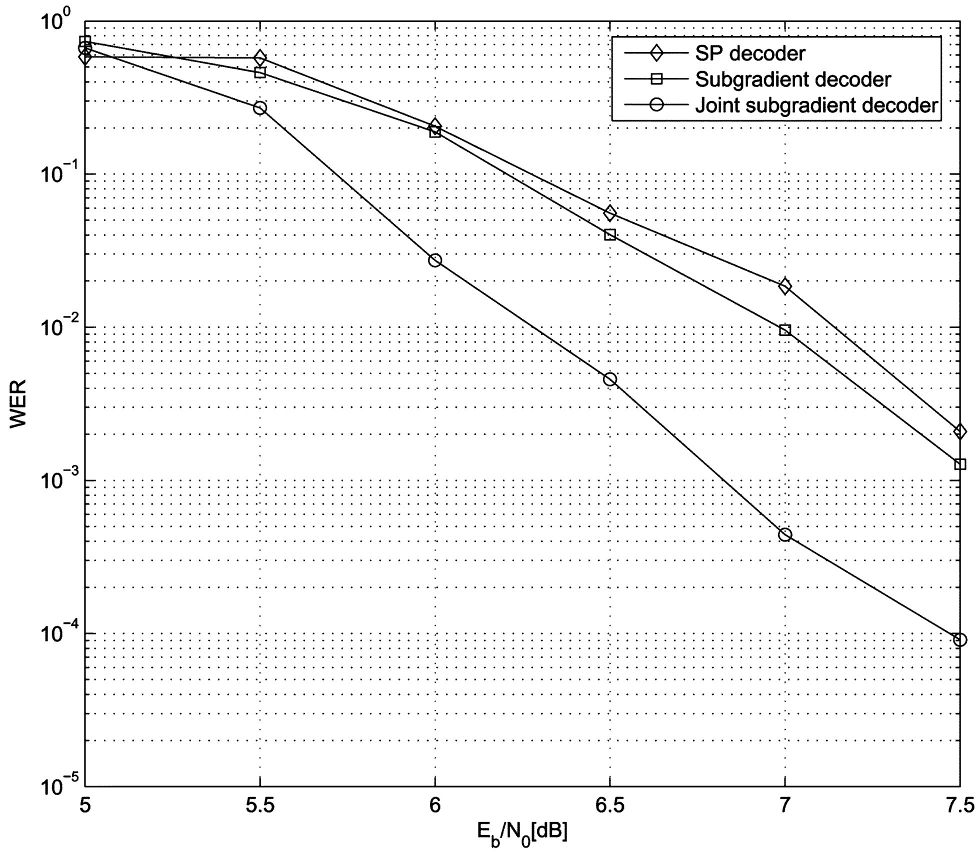

Fig. 7. WER performance of the sum–product decoder, subgradient decoder, and joint subgradient decoder. The number of iterations for the sum–product decoder and subgradient decoder is set as 10. A $(10, 3)$ SPC product code is used. The code length is 1000 and rate is $\left(\frac{9}{10}\right)^3 \approx 0.729$. The average WER is obtained by counting 100 decoding errors in the simulations.

Simulation: The convergence behavior of the proposed subgradient decoder for a $(4, 4)$ SPC product code is shown in Fig. 3. The word error rate (WER) performance of the subgradient decoder with different number of iterations for a $(5, 2)$ SPC product code is shown in Fig. 4. The performance of the subgradient decoder, the joint subgradient decoder, the LP decoder, and the corresponding ML lower bound are illustrated in Figs. 5 and 6 for a $(5, 2)$ code and a $(4, 4)$ code, respectively. The number of iterations is set as 20 for all decoders. It is seen that the LP decoder can practically achieve the ML lower bound. The performance of the simple subgradient decoder incurs loss of a fractional of a decibel with respect to that of the LP decoder. Moreover, the joint subgradient decoder can achieve the ML decoding performance at high SNR at the expense of a higher complexity. The performance of the subgradient and joint subgradient decoders are compared against the sum–product decoder in Fig. 7 for a $(10, 3)$ code. For fair comparison, we use a lookup table to approximate the $\tanh$ function in the sum–product decoder. Thus, the sum–product decoder employed here has a similar complexity of the proposed subgradient decoder. The number of iterations for the subgradient and the sum–product decoders is set as 10. It is seen that the subgradient decoder provides slightly better performance than the sum–product decoder, while the joint subgradient decoder significantly outperforms the sum–product decoder.

V. Conclusion

We have proposed a new LP formulation for the decoding of general linear block codes. The new formulation is much simpler than the original one proposed in [1]. The new formulation offers new insight into the structure of general linear block codes. In particular, we have proved that, for the family of SPC product codes, the fractional distance, BIAWGN, BSC, and BEC pseudodistance are equal to the minimum distance. Moreover, we have developed efficient ML decoders by exploiting the code structure and our new LP formulation for SPC product codes. This is the first class of practical codes that have been shown to asymptotically achieve the ML decoding performance by using methods inspired by LP decoding. Finally, we remark that the proposed new LP formulation may open up new avenues to the analysis of other structured block codes as well as the design of the corresponding efficient decoding algorithms.

References

[1] J. Feldman, M. J. Wainwright, and D. R. Karger, "Using linear programming to decode binary linear codes," *IEEE Trans. Inf. Theory*, vol. 51, no. 1, pp. 954–972, Jan. 2005.

[2] J. Feldman, "Decoding error-correcting codes via linear programming," Ph.D. dissertation, Dept. Electr. Eng. Comput. Sci., Massachusetts Inst. Technol., Cambridge, MA, 2003.

[3] J. Feldman, D. R. Karger, and M. J. Wainwright, "Using linear programming-based decoding of turbo-like codes and its relation to iterative approaches," in *Proc. 40th Annu. Allerton Conf. Commun. Control Comput.*, Allerton, IL, Oct. 2002, pp. 250–260.

[4] R. Koetter and P. O. Vontobel, "Graph covers and iterative decoding of finite-length codes," in *Proc. 3rd Int. Symp. Turbo Codes Related Topics*, Brest, France, Sep. 2003, pp. 75–82.

[5] P. O. Vontobel and R. Koetter, "Graph-cover decoding and finite-length analysis of message-passing iterative decoding of LDPC codes," *IEEE Trans. Inf. Theory*, submitted for publication.

[6] P. O. Vontobel and R. Koetter, "On the relationship between linear programming decoding and min-sum algorithm decoding," in *Proc. Int. Symp. Inf. Theory Appl.*, Parma, Italy, Oct. 2004, pp. 991–996.

[7] P. Chaichanavong and P. H. Siegel, "Relaxation bounds on the minimum pseudo-weight of linear block codes," in *Proc. IEEE Int. Symp. Inf. Theory*, Adelaide, Australia, Sep. 2005, pp. 805–809.

[8] K. Yang, J. Feldman, and X. Wang, "Non-linear programming approaches to decoding low-density parity-check codes," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, pp. 1603–1613, Aug. 2006.

[9] P. O. Vontobel and R. Koetter, "Toward low-complexity linear-programming decoding," in *Proc. 4th Int. Symp. Turbo Codes Related Topics*, Munich, Germany, Apr. 2006, pp. 1603–1613.

[10] M. H. Taghavi and P. Siegel, "Adaptive linear programming decoding," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2006, pp. 1374–1378.

[11] F. R. Kschischang, B. Frey, and H. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[12] M. Chertkov and M. Stepanov, "Pseudo-codeword landscape," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2007, pp. 127–131.

[13] R. Smarandache and P. Vontobel, "Pseudocodeword analysis of tanner graphs from projective and euclidean planes," *IEEE Trans. Inf. Theory*, submitted for publication.

[14] W. Cook, W. Cunningham, W. Pulleyblank, and A. Schrijver, *Combinatorial Optimization*. New York: Wiley, 1998.

[15] D. M. Rankin and T. A. Gulliver, "Single parity-check product codes," *IEEE Trans. Commun.*, vol. 49, no. 8, pp. 1354–1362, Aug. 2001.

[16] D. M. Rankin, T. A. Gulliver, and D. P. Taylor, "Asymptotic performance of single parity-check product codes," *IEEE Trans. Inf. Theory*, vol. 49, no. 9, pp. 2230–2235, Sep. 2003.

[17] P. O. Vontobel, R. Smarandache, N. Kiyavash, J. Teutsch, and D. Vukobratovic, "On the minimal pseudo-codewords of codes from finite geometries," in *Proc. IEEE Int. Symp. Inf. Theory*, Sep. 2005, pp. 980–984.

[18] D. Bertsimas and J. N. Tsitsiklis, *Linear Optimization*. Belmont, MA: Athena Scientific, 1997.

[19] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed. Belmont, MA: Athena Scientific, 1999.

[20] J. Snyders and Y. Be'ery, "Maximum likelihood soft decoding of binary block codes and decoders for the golay codes," *IEEE Trans. Inf. Theory*, vol. IT-35, no. 5, pp. 963–975, Sep. 1989.