A Factor-Graph-Based Random Walk, and its Relevance for LP Decoding Analysis and Bethe Entropy Characterization

Pascal O. Vontobel
Hewlett–Packard Laboratories
Palo Alto, CA 94304, USA
pascal.vontobel@ieee.org

***Abstract*—Although min-sum algorithm (MSA) and linear programming (LP) decoding are tightly related, it is not straightforward to translate MSA decoding performance analysis techniques to the LP decoding setup. Towards closing this performance analysis techniques gap, Koetter and Vontobel [ITAW 2006] showed how the collection of messages from several MSA decoding iterations can be used to construct a dual witness for LP decoding, thereby deriving some performance results for LP decoding.**

In a recent breakthrough paper by Arora, Daskalakis, and Steurer (ADS) [STOC 2009], the understanding of the performance of LP decoding was brought to a new level, not only from the perspective of available analysis tools but also from the perspective of significantly improving the known asymptotic LP decoding threshold bounds. ADS achieved this by showing how MSA decoding analysis type results can be used in the primal domain of the LP decoder, along the way also giving evidence that the above detour over the dual domain is neither necessary nor simpler.

In the present paper we focus on the geometrical aspects of the ADS paper and show that one of the key results of the ADS paper can be reformulated as the construction of a rather nontrivial class of supersets of the fundamental cone, where these supersets are convex cones that are generated by vectors that are derived from computation trees and minimal valid deviations therein. As we will discuss, the main ingredient that allows the verification of this superset construction is a certain class of backtrackless random walks on the code's normal factor graph. Moreover, formulating our results in terms of normal factor graphs will facilitate the generalization of the geometrical results of the ADS paper to setups with non-uniform node degrees, with other types of constraint function nodes, and with no restrictions on the girth. We conclude the paper by showing connections between the entropy rates of the above-mentioned random walks and the Bethe entropy function of the normal factor graph that these random walks are defined on.

I. Introduction

Linear programming (LP) decoding was introduced by Feldman, Wainwright, and Karger [1], [2] as a relaxation of an LP formulation of the blockwise maximum likelihood decoder. Namely, consider a binary linear code $\mathcal{C}$ that is described by some $m \times n$ parity-check matrix $\boldsymbol{H}$. LP decoding can then be written as

$$\boldsymbol{\omega}^{\mathrm{LP}} \triangleq \arg\max_{\boldsymbol{\omega} \in \mathcal{P}} \langle \boldsymbol{\lambda}, \boldsymbol{\omega} \rangle,$$

where $\boldsymbol{\lambda}$ is the length-n vector containing the log-likelihood ratios of the channel output symbols, and where $\mathcal{P} \triangleq \mathcal{P}(\boldsymbol{H})$ is the fundamental polytope, which is a certain relaxation of the codeword polytope $\mathrm{conv}(\mathcal{C})$, which in turn is obtained by embedding $\mathcal{C}$ in $\mathbb{R}^n$ and then taking its convex hull [1], [2], [3], [4]. In the following, we will also need the fundamental cone $\mathcal{K} \triangleq \mathcal{K}(\boldsymbol{H})$, which is the conic hull of $\mathcal{P}$, i.e., $\mathcal{K} \triangleq \mathrm{conic}(\mathcal{P})$, cf. Figure 1 (left).

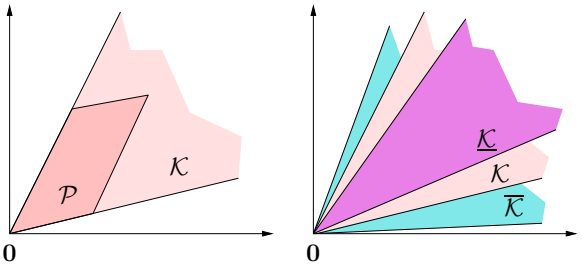

Fig. 1. Left: Fundamental polytope $\mathcal{P}$ and fundamental cone $\mathcal{K}$ of a binary linear code $\mathcal{C}$ that is described by some parity-check matrix $\boldsymbol{H}$. Right: A convex cone $\underline{\mathcal{K}}$ that is a subset of the fundamental cone $\mathcal{K}$, and a convex cone $\overline{\mathcal{K}}$ that is a superset of the fundamental cone $\mathcal{K}$: $\underline{\mathcal{K}} \subseteq \mathcal{K} \subseteq \overline{\mathcal{K}}$.

Assessing the decoding performance of the LP decoder is an intriguing problem. Under the assumption that the channel is a memoryless binary-input output-symmetric channel, it turns out to be sufficient to study the LP decoding performance when the all-zero codeword was sent [1], [2]. Moreover, because of the relationship

$$\left\{ \boldsymbol{\lambda} \;\middle|\; \mathbf{0} \neq \arg\min_{\boldsymbol{\omega} \in \mathcal{P}} \langle \boldsymbol{\lambda}, \boldsymbol{\omega} \rangle \right\} = \left\{ \boldsymbol{\lambda} \;\middle|\; \mathbf{0} \neq \arg\min_{\boldsymbol{\omega} \in \mathcal{K}} \langle \boldsymbol{\lambda}, \boldsymbol{\omega} \rangle \right\},$$

studying if the all-zero codeword loses against any vector in $\mathcal{P}$ is equivalent to studying if the all-zero codeword loses against any vector in $\mathcal{K}$.[1]

Although the determination of the exact decoding error probability of the LP decoder is highly desirable, this is not always feasible because of complexity reasons. This motivates the study of lower and upper bounds on the decoding error

[1]Here and in the following, we assume that the resolution of ties is done in a systematic and/or uniform way. In particular, the resolution of ties can be done in such a way that the error probability under LP decoding when sending the all-zero vector equals the error probability under LP decoding when sending any other codeword. For simplicity of exposition, however, in this text we assume that there are no ties and that the above $\arg\min$ gives back a single vector and not a set with possibly more than one vector.

probability of the LP decoder. A way to obtain such bounds is as follows. Namely, let $\underline{\mathcal{K}}$ and $\overline{\mathcal{K}}$ be convex cones that satisfy

$$\underline{\mathcal{K}} \subseteq \mathcal{K} \subseteq \overline{\mathcal{K}},$$

cf. Figure 1 (right). Then, because of the relationships

$$\left\{\boldsymbol{\lambda} \;\middle|\; \mathbf{0} \neq \arg\min_{\boldsymbol{\omega}\in\mathcal{K}}\langle\boldsymbol{\lambda},\boldsymbol{\omega}\rangle\right\} \supseteq \left\{\boldsymbol{\lambda} \;\middle|\; \mathbf{0} \neq \arg\min_{\boldsymbol{\omega}\in\underline{\mathcal{K}}}\langle\boldsymbol{\lambda},\boldsymbol{\omega}\rangle\right\},$$
$$\left\{\boldsymbol{\lambda} \;\middle|\; \mathbf{0} \neq \arg\min_{\boldsymbol{\omega}\in\mathcal{K}}\langle\boldsymbol{\lambda},\boldsymbol{\omega}\rangle\right\} \subseteq \left\{\boldsymbol{\lambda} \;\middle|\; \mathbf{0} \neq \arg\min_{\boldsymbol{\omega}\in\overline{\mathcal{K}}}\langle\boldsymbol{\lambda},\boldsymbol{\omega}\rangle\right\},$$

it is straightforward to derive the following lower and upper bounds on the decoding error probability of LP decoding

$$\Pr\left(\mathbf{0} \neq \arg\min_{\boldsymbol{\omega}\in\mathcal{P}}\langle\boldsymbol{\Lambda},\boldsymbol{\omega}\rangle\right) \geqslant \Pr\left(\mathbf{0} \neq \arg\min_{\boldsymbol{\omega}\in\underline{\mathcal{K}}}\langle\boldsymbol{\Lambda},\boldsymbol{\omega}\rangle\right), \quad (1)$$

$$\Pr\left(\mathbf{0} \neq \arg\min_{\boldsymbol{\omega}\in\mathcal{P}}\langle\boldsymbol{\Lambda},\boldsymbol{\omega}\rangle\right) \leqslant \Pr\left(\mathbf{0} \neq \arg\min_{\boldsymbol{\omega}\in\overline{\mathcal{K}}}\langle\boldsymbol{\Lambda},\boldsymbol{\omega}\rangle\right), \quad (2)$$

where $\boldsymbol{\Lambda}$ is the random vector associated with the log-likelihood ratio vector $\boldsymbol{\lambda}$.

The inequality in (1) was for example used in [5] to obtain lower bounds on the error probability of the LP decoder. There the convex cone $\underline{\mathcal{K}} \subseteq \mathcal{K}$ was implicitly defined by constructing pseudo-codewords based on a modified version of the canonical completion technique [3], [4].

The inequality in (2) is the focus of this paper, i.e., we want to construct convex cones $\overline{\mathcal{K}} \supseteq \mathcal{K}$ which can be used to obtain upper bounds on the error probability of the LP decoder. Clearly, such cones $\overline{\mathcal{K}}$ are relaxations of the conic hull of the codebook $\mathrm{conic}(\mathcal{C})$ because $\overline{\mathcal{K}}$ is a relaxations of the fundamental cone $\mathcal{K}$, which in turn is a relaxation of the conic hull of the codebook.

There have been several attempts at studying upper bounds on the decoding error probability of LP decoding, in particular towards the goal of formulating asymptotic results. (For example, if the channel is the binary symmetric channel (BSC) then such asymptotic results typically give guarantees that a certain fraction of bit flips can be successfully corrected, either with absolute certainty or with high probability.) In that line of research, let us mention [6], [7] that gave threshold results for low-density parity-check (LDPC) codes whose Tanner graphs [8] have good expansion. Similar type of results were also given in [9], [10]. However, the approach in these latter two papers was to leverage performance analysis tools from message-passing iterative (MPI) decoding, in particular from min-sum algorithm (MSA) decoding. Although all four papers [6], [7], [9], [10] show that LP decoding can correct a constant fraction of errors for the BSC under rather mild conditions on the LDPC code, the analysis techniques in the latter two papers yield much better lower bounds on the fraction of errors that can be corrected. (We refer to [10] for a comparison of some numerical results.)

It is clear that MSA decoding and LP decoding are tightly related [11]. This fact was the motivation for [9] to obtain performance analysis results for LP decoding by leveraging performance analysis results for MSA decoding, in particular by leveraging computation tree based techniques [12]. The main obstacle that has to be overcome when trying to connect the performance analysis of these two decoders is to find a way to merge "locally valid configurations" so as to obtain "globally valid configurations," in the sense that one has to find a way to piece together valid computation tree deviations to form valid configurations in the factor graph [13], [14], [15] that represents the LP decoder.[2] Towards achieving this, [9] considered the LP decoder in the dual domain and pieced together messages from computation trees to form valid configurations on the factor graph that represents the dual of the LP decoder.

Arora, Daskalakis, and Steurer (ADS) [10] had the insight how to achieve a connection between valid computation tree deviations (more precisely, minimal valid computation tree deviations) and valid configurations on the factor graph that represents the LP decoder in the primal domain. Compared to [9], not only is the resulting technique simpler but it also seems to be more powerful. Assuming families of regular LDPC codes with Tanner graphs whose girth grows sufficiently fast, ADS were able to show very good numerical threshold results for LP decoding for the BSC. Interesting extensions of the technique of the ADS paper to memoryless binary-input output-symmetric channels beyond the BSC have recently been presented by Halabi and Even [16].

The LP decoding performance analysis technique in the ADS paper can be seen as having two parts: a geometrical part and a density evolution part.

- **Geometrical part**: The ADS paper implicitly constructs a convex cone $\overline{\mathcal{K}} \supseteq \mathcal{K}$ based on minimal valid computation trees deviations. The present paper will focus entirely on this geometrical aspect of the ADS paper, and will show how the results in the ADS paper can be extended to a much larger family of factor graphs than the one containing only factor graphs of regular LDPC codes. In particular, our results are independent of the girth of the factor graph.
- **Density evolution part**: Once the geometrical part has been established, techniques akin to density evolution [17] can be applied to obtain thresholds and other results. For this part of the LP decoding performance analysis, it remains to be seen how far the requirements on the growth of the girth with respect to the growth of the blocklength can be relaxed. (The threshold results in the ADS paper are based on the girth growing logarithmically with the blocklength.[3])

[2] Valid deviations in computation trees are valid configurations where the assignment at the root is non-zero, see, e.g., Figure 3 in Section V. (Note that for a given factor graph there is a computation tree for every node and for every iteration of the MSA decoding algorithm.)

[3] Note that a randomly constructed Tanner graph with fixed uniform bit node degree and fixed uniform check node degree will *not* have logarithmically growing girth. However, there are explicit deterministic constructions that yield logarithmically growing girth, for example the construction presented in Gallager's thesis [18, Appendix C]. Note that in the context of MSA decoding performance analysis, one usually uses the fact that the fraction of short cycles vanishes asymptotically with high probability [17].

As we will see, the key ingredient to obtain the above-mentioned geometrical results is a class of backtrackless random walks that are defined on normal factor graphs. This class of random walks is such that for every (edge-based) pseudo-codeword there is at least one random walk in this class with the property that the edge visiting probability distribution is proportional to this (edge-based) pseudo-codeword.

Any random walk in this class can be described by a Markov chain with a suitably defined transition probability matrix. Necessarily, the stationary distribution of this Markov chain is an eigenvector (with eigenvalue 1) for the transition probability matrix. Because any eigenvector of the transition probability matrix is "self-consistent", i.e., it is proportional to itself after multiplication by the transition probability matrix, also the stationary distribution vector is "self-consistent." It turns out that this "self-consistency property" of stationary distributions is the crucial ingredient in the verification of the fact that any valid configuration in the LP decoding normal factor graph can be obtained by a suitably weighted combination of valid computation tree deviations. In other words, this "self-consistency property" of stationary distributions guarantees that configurations, although obtained by combining "only locally valid configurations," are "globally valid configurations."

The importance of this class of random walks is corroborated by the fact that this class also appears in the analysis of cycle codes,[4] in particular it gives the link between the Bethe entropy function and the edge zeta function associated with a normal graph [19]. Moreover, these random walks also fit in the theme of expressing a code in terms of some cycle code, along with some additional constraints [20, Section 6].

This paper is structured as follows. Section II collects notations that will be used throughout the paper. Afterwards, Section III defines and discusses a variety of normal graphs, Section IV presents the above-mentioned class of random walks, Section V shows how to construct convex cones that are supersets of the fundamental cone, and Section VI presents some connections between the entropy rates of the above-mentioned random walks and the Bethe entropy function of the normal factor graph that these random walks are defined on. Finally, Section VII contains some conclusions.

Because of space restrictions, Section VI is omitted and proofs are sketched or omitted. All details are provided in the journal version of this paper [21].

II. Notation

This section discusses the most important notations that we will use in this paper. More notational definitions will be given in later sections.

We start with some sets, rings, and fields. We let $\mathbb{Z}$, $\mathbb{Z}_{\geqslant 0}$, $\mathbb{Z}_{>0}$, $\mathbb{R}$, $\mathbb{R}_{\geqslant 0}$, and $\mathbb{R}_{>0}$ be the ring of integers, the set of non-negative integers, the set of positive integers, the field of real numbers, the set of non-negative real numbers, and the set of positive real numbers, respectively. We let $\mathbb{F}_2 \triangleq \{0, 1\}$ be the Galois field with two elements; as a set, $\mathbb{F}_2$ is considered to be a subset of $\mathbb{R}$. The size of a set $\mathcal{S}$ is denoted by $|\mathcal{S}|$.

In the following, all scalars, all entries of vectors, and all entries of matrices will be considered to be in $\mathbb{R}$, unless noted otherwise. So, if an addition or a multiplication is not in the real field, we indicate this, e.g., by writing $a + b$ (in $\mathbb{F}_2$) or $\boldsymbol{a} + \boldsymbol{b}$ (in $\mathbb{F}_2$). As usually done in coding theory, we use only row vectors. The transpose of a vector $\boldsymbol{a}$ is denoted by $\boldsymbol{a}^{\mathsf{T}}$. An inequality of the form $\boldsymbol{a} \geqslant \boldsymbol{b}$ involving two vectors of length N is to be understood component-wise, i.e., $a_i \geqslant b_i$ for all $1 \leqslant i \leqslant N$. We let $\mathbf{0}_N$ and $\mathbf{1}_N$ be, respectively, the all-zero and the all-one row-vector of length N; when the length of these vectors is obvious from the context, we omit the subscript. The support $\mathrm{supp}(\boldsymbol{a})$ of a vector $\boldsymbol{a}$ is the set of indices where $\boldsymbol{a}$ is non-zero. In that context, we use the shorthand $\boldsymbol{a}' \subseteq \boldsymbol{a}$ to denote the statement $\mathrm{supp}(\boldsymbol{a}') \subseteq \mathrm{supp}(\boldsymbol{a})$, i.e., the statement that, for all i, a'_i is non-zero only if a_i is non-zero.

By $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \triangleq \sum_i x_i y_i$ we denote the standard inner product of two vectors having the same length. The ℓ_1-norm of a vector $\boldsymbol{x}$ is $\|\boldsymbol{x}\|_1 \triangleq \sum_i |x_i|$; note that $\|\boldsymbol{x}\|_1 = \langle \boldsymbol{x}, \mathbf{1} \rangle$ if and only if $\boldsymbol{x} \geqslant \mathbf{0}$. Let $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{F}_2^N$ be two vectors of length N. The Hamming weight $w_{\mathrm{H}}(\boldsymbol{x})$ of a vector $\boldsymbol{x}$ is defined to be the number of non-zero positions of $\boldsymbol{x}$ and the Hamming distance $d_{\mathrm{H}}(\boldsymbol{x}, \boldsymbol{y})$ between two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ is defined to be the number of positions where $\boldsymbol{x}$ and $\boldsymbol{y}$ disagree.

We also need some notions from convex geometry (see, e.g., [22]). Let $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(\ell)}$ be ℓ points in $\mathbb{R}^N$. A point of the form $\theta_1 \boldsymbol{x}^{(1)} + \cdots + \theta_\ell \boldsymbol{x}^{(\ell)}$, with $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_\ell)$ such that $\langle \boldsymbol{\theta}, \mathbf{1} \rangle = 1$ and $\boldsymbol{\theta} \geqslant \mathbf{0}$, is called a convex combination of $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(\ell)}$. A set $\mathcal{S} \subseteq \mathbb{R}^N$ is called convex if every possible convex combination of two points of $\mathcal{S}$ is in $\mathcal{S}$. By $\mathrm{conv}(\mathcal{S})$ we denote the convex hull of the set $\mathcal{S}$, i.e., the set that consists of all possible convex combinations of all the points in $\mathcal{S}$; equivalently, $\mathrm{conv}(\mathcal{S})$ is the smallest convex set that contains $\mathcal{S}$.

Again, let $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(\ell)}$ be ℓ points in $\mathbb{R}^N$. A point of the form $\theta_1 \boldsymbol{x}^{(1)} + \cdots + \theta_\ell \boldsymbol{x}^{(\ell)}$ with $\boldsymbol{\theta} \geqslant \mathbf{0}$ is called a conic combination of $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(\ell)}$. A set $\mathcal{K} \subseteq \mathbb{R}^N$ is called a convex cone if every possible conic combination of two points of $\mathcal{K}$ is in $\mathcal{K}$. By $\mathrm{conic}(\mathcal{S})$ we denote the conic hull of the set $\mathcal{S}$, i.e., the set that consists of all possible conic combinations of all the points in $\mathcal{S}$; equivalently, $\mathrm{conic}(\mathcal{S})$ is the smallest convex conic set that contains $\mathcal{S}$.

Finally, we use Iverson's convention, i.e., for a statement S we define $[S] \triangleq 1$ if S is true and $[S] \triangleq 0$ otherwise.

III. Normal Graphs and the Local Marginal Polytope

We will express our results in terms of normal graphs [14], which are also known as normal factor graphs or Forney-style factor graphs.

Definition 1: A normal graph $\mathsf{N}(\mathcal{F}, \mathcal{E}, \mathcal{A}, \mathcal{G})$ consists of

- a graph $(\mathcal{F}, \mathcal{E})$, with vertex set $\mathcal{F}$ (also known as function node set $\mathcal{F}$) and (half-)edge set $\mathcal{E}$.
- a collection of alphabets $\mathcal{A} \triangleq \{\mathcal{A}_e\}_{e \in \mathcal{E}}$, where the alphabet $\mathcal{A}_e$ is associated with the edge $e \in \mathcal{E}$;

[4] Cycle codes are LDPC codes described by a parity-check matrix with uniform column weight two.

- a collection of functions $\mathcal{G} \triangleq \{g_f\}_{f\in\mathcal{F}}$, where the local function g_f is associated with the function node $f \in \mathcal{F}$ and further specified below. □

In the following, we will, for every $f \in \mathcal{F}$, use $\mathcal{E}_f \subseteq \mathcal{E}$ to denote the subset of edges that is incident to f, and for a vector $\boldsymbol{c} \in \prod_{e\in\mathcal{E}} \mathcal{A}_e$ we define for every $f \in \mathcal{F}$ the vector $\boldsymbol{c}_f \triangleq (c_e)_{e\in\mathcal{E}_f}$.

With this, for a multiplicatively written normal graph the global function $g : \prod_{e\in\mathcal{E}} \mathcal{A}_e \to \mathbb{R}$ is defined to be $g(\boldsymbol{c}) \triangleq \prod_{f\in\mathcal{F}} g_f(\boldsymbol{c}_f)$ with local functions $g_f : \left(\prod_{e\in\mathcal{E}_f} \mathcal{A}_e\right) \to \mathbb{R}$, $f \in \mathcal{F}$, whereas for an additively written normal graph (that typically represents some type of cost function) the global function $g : \prod_{e\in\mathcal{E}} \mathcal{A}_e \to \mathbb{R} \cup \{\infty\}$ is defined to be $g(\boldsymbol{c}) \triangleq \sum_{f\in\mathcal{F}} g_f(\boldsymbol{c}_f)$ with local functions $g_f : \left(\prod_{e\in\mathcal{E}_f} \mathcal{A}_e\right) \to \mathbb{R} \cup \{\infty\}$, $f \in \mathcal{F}$.

For a multiplicatively written normal graph, we define for every $f \in \mathcal{F}$ the function node alphabet $\mathcal{A}_f$ to be the set

$$\mathcal{A}_f \triangleq \left\{ \boldsymbol{a}_f \in \prod_{e\in\mathcal{E}_f} \mathcal{A}_e \;\middle|\; g_f(\boldsymbol{a}_f) \neq 0 \right\},$$

and for an additively written normal graph we define for every $f \in \mathcal{F}$ the function node alphabet $\mathcal{A}_f$ to be

$$\mathcal{A}_f \triangleq \left\{ \boldsymbol{a}_f \in \prod_{e\in\mathcal{E}_f} \mathcal{A}_e \;\middle|\; g_f(\boldsymbol{a}_f) \neq \infty \right\}.$$

The alphabets $\mathcal{A}_f$, $f \in \mathcal{F}$, will also be considered to be part of the collection $\mathcal{A}$.

In the following, we will use $a_{f,e}$ to denote the component of $\boldsymbol{a}_f$ related to the edge $e \in \mathcal{E}_f$, we will use the short-hand $\sum_{\boldsymbol{a}_f}$ for $\sum_{\boldsymbol{a}_f\in\mathcal{A}_f}$, and we will use the short-hand $\sum_{a_e}$ for $\sum_{a_e\in\mathcal{A}_e}$.

Finally, a vector $\boldsymbol{c} \in \prod_{e\in\mathcal{E}} \mathcal{A}_e$ will be called a configuration of the normal graph, and a configuration $\boldsymbol{c}$ with $g(\boldsymbol{c}) \neq 0$ (with $g(\boldsymbol{c}) \neq \infty$ in the context of additively normal graphs) will be called a valid configuration. Clearly, the set of valid configurations $\mathcal{C}_{\text{edge}}$ is characterized as follows

$$\mathcal{C}_{\text{edge}} \triangleq \left\{ (c_e)_{e\in\mathcal{E}} \;\middle|\; \begin{array}{l} c_e \in \mathcal{A}_e \text{ for all } e \in \mathcal{E} \\ \boldsymbol{c}_f \in \mathcal{A}_f \text{ for all } f \in \mathcal{F} \end{array} \right\}.$$

In this paper we will focus on a special class of normal graphs as defined below.

Definition 2: Let $\mathcal{N}$ be the collection of all normal graphs $\mathsf{N}(\mathcal{F}, \mathcal{E}, \mathcal{A}, \mathcal{G})$

- where $|\mathcal{F}| < \infty$ and $|\mathcal{E}| < \infty$,
- where $\mathcal{E}$ contains no half-edges,
- where $\mathcal{A}_e = \{0, 1\}$ for all $e \in \mathcal{E}$, and
- where $w_{\text{H}}(\boldsymbol{a}_f) \neq 1$ for all $f \in \mathcal{F}$, $\boldsymbol{a}_f \in \mathcal{A}_f$. □

Let us comment on this definition.

- The first constraint is not much of a constraint since usually we are interested in finite graphs.
- Also the second constraint is not really much of a constraint since any normal graph with half-edges can be turned into another normal factor graph where the variables associated with the half-edges are "marginalized out" by modifying the adjacent function nodes. (Here the marginalization process depends on the type of message-passing algorithm that is applied to the normal graph.)
- Moreover, note that any normal graph with a degree-one function node can also be turned into a normal graph without this degree-one function node. Namely, let f be such a degree-one function node and let e be the edge between the function node f and some other function node f'. Then, "marginalizing out" over the variable associated with e and over the function node f, we obtain a new normal graph without edge e, without function node f, and with a modified function node f'. Applying this procedure repeatedly if necessary, we obtain the "core" of the normal graph that contains only function nodes of degree at least two.
- A class of normal graphs that is not included in $\mathcal{N}$ (even after the above-mentioned graph modifications) is the class of normal graphs that have function nodes whose degree is at least two and whose alphabet contains elements of weight one. However, in coding theory such function nodes usually do not appear since normal graphs with such function nodes do not yield good codes.

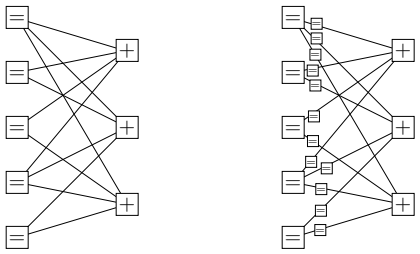

Fig. 2. Left: Normal graph for Example 3. Right: Normal graph for Example 5.

Example 3: Consider a binary linear code $\mathcal{C}$ of length n described by a parity-check matrix $\boldsymbol{H} \in \mathbb{F}_2^{m\times n}$, i.e.,

$$\mathcal{C} \triangleq \left\{ \boldsymbol{x} \in \mathbb{F}_2^n \;\middle|\; \boldsymbol{H} \cdot \boldsymbol{x}^{\mathsf{T}} = \boldsymbol{0}^{\mathsf{T}} \right\}.$$

In the same way that we can draw a Tanner graph for this code, we can draw a normal graph whose global function represents the indicator function of the code. Let the set of function nodes be $\mathcal{F} \triangleq \mathcal{I} \cup \mathcal{J}$, where $\mathcal{I}$ is the set of all column indices of $\boldsymbol{H}$ and $\mathcal{J}$ is the set of all row indices of $\boldsymbol{H}$, and let the set of edges be $\mathcal{E} \triangleq \left\{(i, j) \in \mathcal{I} \times \mathcal{J} \;\middle|\; h_{j,i} = 1\right\}$. If the function node f is in $\mathcal{I}$, then g_f is defined to be an equal function node of degree $|\mathcal{E}_f|$, i.e.,

$$\begin{aligned} \mathcal{A}_f &= \left\{ \boldsymbol{a}_f \in \{0,1\}^{|\mathcal{E}_f|} \;\middle|\; w_{\text{H}}(\boldsymbol{a}_f) \in \left\{0, |\mathcal{E}_f|\right\} \right\}, \\ g_f(\boldsymbol{a}_f) &= \left[\boldsymbol{a}_f \in \mathcal{A}_f\right]. \end{aligned}$$

If the function node f is in $\mathcal{J}$, then g_f is defined to be a single parity-check function node of degree $|\mathcal{E}_f|$, i.e.,

$$\begin{aligned} \mathcal{A}_f &= \left\{ \boldsymbol{a}_f \in \{0,1\}^{|\mathcal{E}_f|} \;\middle|\; w_{\text{H}}(\boldsymbol{a}_f) \text{ is even} \right\}, \\ g_f(\boldsymbol{a}_f) &= \left[\boldsymbol{a}_f \in \mathcal{A}_f\right]. \end{aligned}$$

For example, the parity-check matrix

$$\boldsymbol{H} \triangleq \begin{pmatrix} 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \end{pmatrix} \tag{3}$$

yields the normal graph shown in Figure 2 (left).

To be precise, the above procedure does strictly speaking not define the indicator function $[\boldsymbol{x} \in \mathcal{C}]$, but the indicator function $[\boldsymbol{c} \in \mathcal{C}_{\text{edge}}]$. However, there is a bijection between codewords $\boldsymbol{x} = (x_i)_{i\in\mathcal{I}} \in \mathcal{C}$ and valid configurations $\boldsymbol{c} = (c_e)_{e\in\mathcal{E}} \in \mathcal{C}_{\text{edge}}$, where $c_e = x_f$ for all $e \in \mathcal{E}_f$, $f \in \mathcal{I}$. □

Note that in the case where the parity-check matrix $\boldsymbol{H}$ in Example 3 contains a column with a single one in it, i.e., there exists an $f \in \mathcal{I}$ such that $|\mathcal{E}_f| = 1$, then the resulting normal graph is not in $\mathcal{N}$.

Example 4: This example continues Example 3. Assume that the code $\mathcal{C}$ is used for data transmission over a binary-input memoryless channel with channel law $W(y|x)$, where x is a channel input symbol and y is a channel output symbol, and that we would like the global function of the normal graph to be proportional to the indicator function of the code times $\prod_{i\in\mathcal{I}} W(y_i|x_i)$.

It is possible to formulate a normal graph in $\mathcal{N}$ with this global function. Namely, starting with the normal graph in Example 3, for every $f \in \mathcal{I}$, the equal function node is replaced by a modified equal function node as follows: the set $\mathcal{A}_f$ is defined as in Example 3 but the function g_f is modified to read $g_f(\boldsymbol{a}_f) = \big[\boldsymbol{a}_f \in \mathcal{A}_f\big] \cdot W(y_f|a_{f,e})$, where e is arbitrary in $\mathcal{E}_f$. Moreover, for every $f \in \mathcal{J}$, the set $\mathcal{A}_f$ and the function g_f are defined as in Example 3. □

Example 5: This example continues Examples 3 and 4. As an alternative to the procedure that modified the normal graph in Example 3 to obtain the normal graph in Example 4, we can also modify the normal graph in Example 3 as follows. Namely, $\mathcal{A}_f$ and g_f are left unchanged for all $f \in \mathcal{I} \cup \mathcal{J}$, but every edge $e \in \mathcal{E}$ is replaced by two edges e' and e'', along with a function node f that is a modified equal function node with incident edges e' and e'' and with

$$\mathcal{A}_f = \big\{\boldsymbol{a}_f \in \{0,1\}^2 \;\big|\; w_{\text{H}}(\boldsymbol{a}_f) \in \{0,2\}\big\},$$
$$g_f(\boldsymbol{a}_f) = \big[a_{f,e'} = a_{f,e''}\big] \cdot \big(W(y_f|a_{f,e'})\big)^{1/|\mathcal{E}_i|},$$

where $i \in \mathcal{I}$ is the column index of $\boldsymbol{H}$ corresponding to the edge e. This approach is exemplified in Figure 2 (right) for the parity-check matrix $\boldsymbol{H}$ shown in (3). □

Given a normal graph $\mathsf{N}(\mathcal{F},\mathcal{E},\mathcal{A},\mathcal{G}) = \mathsf{N}\big(\mathcal{F},\mathcal{E},\{\mathcal{A}_f\}_f \cup \{\mathcal{A}_e\}_e,\mathcal{G}\big)$, the LP relaxation normal graph is defined to be the normal graph $\mathsf{N}^{\text{LP}}\big(\mathcal{F},\mathcal{E},\{\text{conv}(\mathcal{A}_f)\}_f \cup \{\text{conv}(\mathcal{A}_e)\}_e,\mathcal{G}^{\text{LP}}\big)$, where $\mathcal{G}^{\text{LP}}$ is suitably extended from $\mathcal{G}$. (This extension depends on whether N is an additively or a multiplicatively written normal graph. We omit the details.)

The local marginal polytope (see, e.g., [23], [24]), defined next, is tightly related to the set of valid configurations $\mathcal{C}_{\text{edge}}^{\text{LP}}$ of N^{LP}.

Definition 6: Consider a normal graph $\mathsf{N}(\mathcal{F},\mathcal{E},\mathcal{A},\mathcal{G}) \in \mathcal{N}$. Let $\boldsymbol{\beta} \triangleq \big((\boldsymbol{\beta}_f)_{f\in\mathcal{F}}, (\boldsymbol{\beta}_e)_{e\in\mathcal{E}}\big)$ be a collection of vectors based on the real vectors $\boldsymbol{\beta}_f \triangleq (\beta_{f,\boldsymbol{a}_f})_{\boldsymbol{a}_f\in\mathcal{A}_f}$, $\boldsymbol{\beta}_e \triangleq (\beta_{e,a_e})_{a_e\in\mathcal{A}_e}$. Then, for $f \in \mathcal{F}$, the fth local marginal polytope (or fth belief polytope) $\mathcal{B}_f$ is defined to be the set

$$\mathcal{B}_f \triangleq \left\{\boldsymbol{\beta}_f \in \mathbb{R}_{\geqslant 0}^{|\mathcal{A}_f|} \;\middle|\; \sum_{\boldsymbol{a}_f} \beta_{f,\boldsymbol{a}_f} = 1\right\},$$

and for all $e \in \mathcal{E}$, the eth local marginal polytope (or eth belief polytope) $\mathcal{B}_e$ is defined to be the set

$$\mathcal{B}_e \triangleq \left\{\boldsymbol{\beta}_e \in \mathbb{R}_{\geqslant 0}^{|\mathcal{A}_e|} \;\middle|\; \sum_{a_e} \beta_{e,a_e} = 1\right\}.$$

With this, the local marginal polytope (or belief polytope) $\mathcal{B}$ is defined to be the set

$$\mathcal{B} = \left\{ \boldsymbol{\beta} \;\middle|\; \begin{array}{c} \boldsymbol{\beta}_f \in \mathcal{B}_f \text{ for all } f \in \mathcal{F} \\ \boldsymbol{\beta}_e \in \mathcal{B}_e \text{ for all } e \in \mathcal{E} \\ \displaystyle\sum_{\boldsymbol{a}_f \in \mathcal{A}_f:\, a_{f,e} = a_e} \beta_{f,\boldsymbol{a}_f} = \beta_{e,a_e} \\ \text{for all } f \in \mathcal{F},\, e \in \mathcal{E}_f,\, a_e \in \mathcal{A}_e \end{array} \right\},$$

where $\boldsymbol{\beta} \in \mathcal{B}$ is called a pseudo-marginal. (The constraints that were listed last in the definition of $\mathcal{B}$ will be called "edge consistency constraints.") □

Definition 7: Consider a normal graph $\mathsf{N}(\mathcal{F},\mathcal{E},\mathcal{A},\mathcal{G}) \in \mathcal{N}$. We define the edge-based fundamental polytope $\mathcal{P}_{\text{edge}}$ and the edge-based fundamental cone $\mathcal{K}_{\text{edge}}$ to be, respectively,

$$\mathcal{P}_{\text{edge}} \triangleq \big\{(\beta_{e,1})_{e\in\mathcal{E}} \;\big|\; \boldsymbol{\beta} \in \mathcal{B}\big\},$$
$$\mathcal{K}_{\text{edge}} \triangleq \text{conic}(\mathcal{P}_{\text{edge}}).$$

Elements of $\mathcal{P}_{\text{edge}}$ and $\mathcal{K}_{\text{edge}}$ will be called edge-based pseudo-codewords. The connection to N^{LP} is given by $\mathcal{C}_{\text{edge}}^{\text{LP}} = \mathcal{P}_{\text{edge}}$. We will also need the projection

$$\begin{aligned} \boldsymbol{\psi}_{\text{edge}} : \mathcal{B} &\to \mathcal{P}_{\text{edge}}, \\ \boldsymbol{\beta} &\mapsto (\beta_{e,1})_{e\in\mathcal{E}}. \end{aligned}$$

(Clearly, in general there are many $\boldsymbol{\beta} \in \mathcal{B}$ that map to the same edge-based pseudo-codeword in $\mathcal{P}_{\text{edge}}$.) □

The (usual) fundamental polytope $\mathcal{P} \triangleq \mathcal{P}(\boldsymbol{H})$ [1], [2], [3], [4] of some parity-check matrix $\boldsymbol{H}$ representing some code $\mathcal{C}$ is related as follows to the edge-based fundamental polytope $\mathcal{P}_{\text{edge}}$ of the normal graph that is associated with $\boldsymbol{H}$ according to the construction in Example 3. Namely, there is a bijection between pseudo-codewords $\boldsymbol{\omega} = (\omega_i)_{i\in\mathcal{I}} \in \mathcal{P}$ and edge-based pseudo-codewords $\boldsymbol{\epsilon} = (\epsilon_e)_{e\in\mathcal{E}} \in \mathcal{P}_{\text{edge}}$, where $\epsilon_e = \omega_f$ for all $e \in \mathcal{E}_f$, $f \in \mathcal{I}$.

The next object will be crucial towards defining one of the main objects of this paper, namely backtrackless random walks on normal graphs.

Definition 8: Consider a normal graph $\mathsf{N}(\mathcal{F},\mathcal{E},\mathcal{A},\mathcal{G}) \in \mathcal{N}$. Based on this normal graph, we define a new normal graph $\mathring{\mathsf{N}}\big(\mathring{\mathcal{F}},\mathring{\mathcal{E}},\mathring{\mathcal{A}},\mathring{\mathcal{G}}\big) \in \mathcal{N}$ with set of valid configurations $\mathring{\mathcal{C}}_{\text{edge}}$, with local marginal polytope $\mathring{\mathcal{B}}$, with edge-based fundamental polytope $\mathring{\mathcal{P}}_{\text{edge}}$, and with edge-based fundamental cone $\mathring{\mathcal{K}}_{\text{edge}}$ as follows.

- $\mathring{\mathcal{F}} \triangleq \mathcal{F}$ and $\mathring{\mathcal{E}} \triangleq \mathcal{E}$.
- For every $f \in \mathring{\mathcal{F}}$,
$$\mathring{\mathcal{A}}_f \triangleq \left\{ \mathring{\boldsymbol{a}}_f \in \{0,1\}^{|\mathcal{E}_f|} \;\middle|\; w_{\mathrm{H}}(\mathring{\boldsymbol{a}}_f) \in \{0,2\} \right\}.$$
- For every $e \in \mathcal{E}$,
$$\mathring{\mathcal{A}}_e \triangleq \{0,1\}.$$
- The local functions $\mathring{g}_f$, $f \in \mathring{\mathcal{F}}$, are left unspecified, but their respective supports are assumed to match the sets $\mathring{\mathcal{A}}_f$, $f \in \mathring{\mathcal{F}}$.
- $\mathring{\mathcal{C}}_{\mathrm{edge}}$, $\mathring{\mathcal{B}}$, $\mathring{\mathcal{P}}_{\mathrm{edge}}$, $\mathring{\mathcal{K}}_{\mathrm{edge}}$ for $\mathring{\mathsf{N}}$ are defined analogously to $\mathcal{C}_{\mathrm{edge}}$, $\mathcal{B}$, $\mathcal{P}_{\mathrm{edge}}$, $\mathcal{K}_{\mathrm{edge}}$ for N.

Note that the little circle on top of $\mathring{\mathsf{N}}$, etc., is mnemonic for the fact that valid configurations in this new normal graph form cycles, or vertex-disjoint cycles, in the underlying graph; we may therefore call this new normal graph $\mathring{\mathsf{N}}$ a vertex-disjoint cycle normal graph. Note that, unless the degree of all function nodes is two or three, $\mathring{\mathcal{C}}_{\mathrm{edge}}$ is *not* a cycle code. However, one can show that $\mathring{\mathcal{K}}_{\mathrm{edge}}$ equals the edge-based fundamental cone of the cycle code defined on $(\mathcal{F}, \mathcal{E})$. □

The above definition of the normal graph $\mathring{\mathsf{N}}$ based on N can be seen as a distillation of several earlier concepts that proved to be useful.

- Expressing a code in terms of a cycle code, along with some additional constraints, as was done in [20, Section 6].
- A certain function that was very useful in the proof of Lemma 2 in [10, Section 5.1].
- A construction of a new normal graph in [19] based on a normal graph defining a cycle code. In fact, the construction in [19] is a special case of the construction above. (Note that in the case of cycle codes, $\mathring{\mathcal{A}}_f \subseteq \mathcal{A}_f$ for all $f \in \mathcal{F}$.)

The next definition introduces a mapping between certain pseudo-marginals in $\mathcal{B}$ and pseudo-marginals in $\mathring{\mathcal{B}}$. For this we will need the set $\mathcal{O} \triangleq \left\{ \boldsymbol{\epsilon} \in \mathbb{R}^{|\mathcal{E}|}_{\geqslant 0} \;\middle|\; \|\boldsymbol{\epsilon}_f\|_1 \leqslant 2 \text{ for all } f \in \mathcal{F} \right\}$ that is a polytope containing points in $\mathbb{R}^{|\mathcal{E}|}$ that have non-negative coordinates and that are (somewhat) close to the origin. Note that the conic hull of $\mathcal{O}$ equals $\mathbb{R}^{|\mathcal{E}|}_{\geqslant 0}$.

Definition 9: Let $\boldsymbol{\beta} \in \mathcal{B}$ be such that $\boldsymbol{\epsilon} \triangleq \boldsymbol{\psi}_{\mathrm{edge}}(\boldsymbol{\beta}) \in \mathcal{O}$. Then we associate with $\boldsymbol{\beta}$ the pseudo-marginal $\mathring{\boldsymbol{\beta}} \in \mathring{\mathcal{B}}$ as follows.

- For every $f \in \mathcal{F}$,
$$\mathring{\beta}_{f,\mathring{\boldsymbol{a}}_f} \triangleq \begin{cases} 1 - \frac{1}{2} \cdot \|\boldsymbol{\epsilon}_f\|_1 & \text{if } w_{\mathrm{H}}(\mathring{\boldsymbol{a}}_f) = 0 \\ \sum_{\boldsymbol{a}_f} \beta_{f,\boldsymbol{a}_f} \cdot \frac{\left[\mathring{\boldsymbol{a}}_f \subseteq \boldsymbol{a}_f\right]}{w_{\mathrm{H}}(\boldsymbol{a}_f) - 1} & \text{if } w_{\mathrm{H}}(\mathring{\boldsymbol{a}}_f) = 2 \end{cases}.$$
(Note that the term corresponding to $\boldsymbol{a}_f \in \mathcal{A}_f$ contributes to the above sum only if $\beta_{f,\boldsymbol{a}_f} > 0$ and $\mathring{\boldsymbol{a}}_f \subseteq \boldsymbol{a}_f$.)
- For every $e \in \mathcal{E}$,
$$\mathring{\boldsymbol{\beta}}_e \triangleq \boldsymbol{\beta}_e.$$

□

An Appendix showing that $\mathring{\boldsymbol{\beta}}$ in Definition 9 is well defined is omitted.

Let us comment on this definition.

- Let $\boldsymbol{\beta} \in \mathcal{B}$ be such that it has an associated pseudo-marginal $\mathring{\boldsymbol{\beta}} \in \mathring{\mathcal{B}}$ according to the above definition. Introducing, $\boldsymbol{\epsilon} \triangleq \boldsymbol{\psi}_{\mathrm{edge}}(\boldsymbol{\beta})$ and $\mathring{\boldsymbol{\epsilon}} \triangleq \boldsymbol{\psi}_{\mathrm{edge}}(\mathring{\boldsymbol{\beta}})$, we obtain
$$\boldsymbol{\epsilon} = \mathring{\boldsymbol{\epsilon}},$$
i.e., both $\boldsymbol{\beta}$ and $\mathring{\boldsymbol{\beta}}$ yield the same edge-based pseudo-codeword. □
- The above remark implies that $\mathcal{P}_{\mathrm{edge}} \cap \mathcal{O} \subseteq \mathring{\mathcal{P}}_{\mathrm{edge}} \cap \mathcal{O}$, and so $\mathcal{K}_{\mathrm{edge}} \subseteq \mathring{\mathcal{K}}_{\mathrm{edge}}$. Therefore, $\mathring{\mathcal{K}}_{\mathrm{edge}}$ is a superset of $\mathcal{K}_{\mathrm{edge}}$ and can be used to obtain upper bounds on the LP decoding performance. However, $\mathring{\mathcal{K}}_{\mathrm{edge}}$ equals the edge-based fundamental cone of the cycle code defined on $(\mathcal{F}, \mathcal{E})$, and because cycle codes are relatively weak performance-wise, one expects that the resulting bounds would be loose. Nevertheless, the supersets $\overline{\mathcal{K}}_{\mathrm{edge}}$ that will appear in Theorem 21 are related to $\mathring{\mathcal{K}}_{\mathrm{edge}}$. Namely, $\mathcal{K}_{\mathrm{edge}} \subseteq \overline{\mathcal{K}}_{\mathrm{edge}} \subseteq \mathring{\mathcal{K}}_{\mathrm{edge}}$.
- Consider a binary linear code $\mathcal{C}$ defined by some parity-check matrix $\boldsymbol{H}$, along with the normal graph as defined in Example 3. Define the set $\mathcal{H} \triangleq \left\{ \boldsymbol{\epsilon} \in \mathbb{R}^{|\mathcal{E}|} \;\middle|\; \text{for all } f \in \mathcal{I}\text{: } \epsilon_e = \epsilon_{e'} \text{ for all } e, e' \in \mathcal{E}_f \right\}$. It follows that $\mathcal{P}_{\mathrm{edge}} \cap \mathcal{O} = \mathring{\mathcal{P}}_{\mathrm{edge}} \cap \mathcal{O} \cap \mathcal{H}$, and so $\mathcal{K}_{\mathrm{edge}} = \mathring{\mathcal{K}}_{\mathrm{edge}} \cap \mathcal{H}$. This observation generalizes the construction in [20, Section 6] where a code was expressed in terms of a cycle code, along with some additional constraints. (Note that in contrast to [20, Section 6] we do not require the column weights of the parity-check matrix $\boldsymbol{H}$ to be even integers.)

For the rest of this paper we will assume that we consider a fixed normal graph $\mathsf{N}(\mathcal{F}, \mathcal{E}, \mathcal{A}, \mathcal{G}) \in \mathcal{N}$, along with the normal graph $\mathring{\mathsf{N}}\big(\mathring{\mathcal{F}}, \mathring{\mathcal{E}}, \mathring{\mathcal{A}}, \mathring{\mathcal{G}}\big) \in \mathcal{N}$ that was specified in Definition 9. Moreover, $\mathcal{B}$ and $\mathring{\mathcal{B}}$ will be the local marginal polytopes associated with these two normal graphs. Note that $(\mathcal{F}, \mathcal{E}) = (\mathring{\mathcal{F}}, \mathring{\mathcal{E}})$, but $\mathcal{A}$ and $\mathring{\mathcal{A}}$ are in general different.

IV. Associating a Random Walk with a Pseudo-Marginal

We come now to one of the main objects of this paper, namely a certain class of backtrackless random walks on a normal graph. Many of the definitions in this section were motivated by similar definitions in [19], and by some concepts in [10, Section 5.1].

Let us start with some graph-related definitions that will be helpful later on for specifying these backtrackless random walks.

Definition 10 ([25], [20]): Let $(\mathcal{F}, \mathcal{E})$ be some graph, and assume that $\mathcal{E} = \{1, \ldots, |\mathcal{E}|\}$. A *directed graph* derived from $(\mathcal{F}, \mathcal{E})$ is any pair $(\mathcal{F}, \mathcal{D})$ where
$$\mathcal{D} \triangleq \left\{ d_e \;\middle|\; e \in \mathcal{E} \right\} \cup \left\{ d_{|\mathcal{E}|+e} \;\middle|\; e \in \mathcal{E} \right\}$$
is a set of ordered triples $(f, e, f') \in \mathcal{F} \times \mathcal{E} \times \mathcal{F}$ such that, for all $e \in \mathcal{E}$, if e connects f and f', then either
$$d_e \triangleq (f, e, f') \quad \text{and} \quad d_{|\mathcal{E}|+e} \triangleq (f', e, f), \quad \text{or}$$
$$d_e \triangleq (f', e, f) \quad \text{and} \quad d_{|\mathcal{E}|+e} \triangleq (f, e, f').$$

(Thus we may think of $(\mathcal{F},\mathcal{D})$ as having two directed edges, with opposite directions, for every edge of $(\mathcal{F},\mathcal{E})$.)

We will use $e(d)$ to denote the undirected edge in $\mathcal{E}$ that is associated with some directed edge $d \in \mathcal{D}$. Moreover, for every $e \in \mathcal{E}$ the set $\mathcal{D}_e$ will be defined to contain the two directed edges that are associated with e, i.e., $\mathcal{D}_e \triangleq \{d \in \mathcal{D} \mid e(d) = e\}$, and for every $f \in \mathcal{F}$ the set $\mathcal{D}_f$ will contain all the directed edges pointing out of the function node f.

The so-called *directed edge matrix* of $(\mathcal{F},\mathcal{D})$ is the $|\mathcal{D}|\times|\mathcal{D}|$ matrix $\boldsymbol{M} = (m_{d,d'})_{d\in\mathcal{D},d'\in\mathcal{D}}$ with

$$m_{d,d'} = \begin{cases} 1 & \text{if } d=(f_1,e,f_2) \text{ and } d'=(f_1',e',f_2') \\ & \text{are such that } f_2 = f_1' \text{ and } e \neq e' \\ 0 & \text{otherwise} \end{cases}.$$

With this, we define for every $d \in \mathcal{D}$ the set $\mathcal{D}_d \triangleq \left\{d' \in \mathcal{D} \;\middle|\; m_{d,d'} = 1\right\}$, which is the set of directed edges which the directed edge d can feed into. □

Definition 11: For every $\mathring{\boldsymbol{\beta}} \in \mathring{\mathcal{B}}$, we will use the following definitions. Namely, for every $d \in \mathcal{D}$ we define

$$\mathring{\beta}_d \triangleq \frac{1}{2}\mathring{\beta}_{e(d),1},$$

and for every $(d,d') \in \mathcal{D}^2$ we define

$$\mathring{\beta}_{d,d'} \triangleq \begin{cases} \frac{1}{2}\mathring{\beta}_{f,\mathring{\boldsymbol{a}}_f} & \text{if } d' \in \mathcal{D}_d \\ 0 & \text{if } d' \notin \mathcal{D}_d \end{cases},$$

where $f \in \mathcal{F}$ and $\mathring{\boldsymbol{a}}_f \in \mathring{\mathcal{A}}_f$ are such that $\operatorname{supp}(\mathring{\boldsymbol{a}}_f) = \left\{e(d), e(d')\right\} \subseteq \mathcal{E}_f$. □

Lemma 12: With the specifications in Definition 11,

$$\sum_{d'\in\mathcal{D}_d} \mathring{\beta}_{d,d'} = \mathring{\beta}_d \quad \text{for any } d \in \mathcal{D},$$
$$\sum_{d:\, d'\in\mathcal{D}_d} \mathring{\beta}_{d,d'} = \mathring{\beta}_{d'} \quad \text{for any } d' \in \mathcal{D}.$$

Proof: We prove only the first statement; the second statement will follow analogously. Let $d = (f_1,e,f_2) \in \mathcal{D}$. Then

$$\sum_{d'\in\mathcal{D}_d} \mathring{\beta}_{d,d'} = \sum_{\mathring{\boldsymbol{a}}_{f_2}\in\mathring{\mathcal{A}}_{f_2}:\, \mathring{a}_{f_2,e}=1} \frac{1}{2}\mathring{\beta}_{f_2,\mathring{\boldsymbol{a}}_{f_2}} \overset{\text{(a)}}{=} \frac{1}{2}\mathring{\beta}_{e,1} = \mathring{\beta}_d,$$

where at step (a) we have used the fact that $\mathring{\boldsymbol{\beta}} \in \mathring{\mathcal{B}}$ satisfies the edge consistency constraints in $\mathring{\mathcal{B}}$. □

Definition 13: Let $\mathring{\boldsymbol{\beta}} \in \mathring{\mathcal{B}}$. Based on such a $\mathring{\boldsymbol{\beta}}$ we define a time-invariant Markov process with the following properties.

- Its state space is the set of directed edges $\mathcal{D}$.
- The time-invariant transition probability of going from state $d \in \mathcal{D}$ to state $d' \in \mathcal{D}$ is defined to be

$$\mathring{p}_{d,d'} \triangleq \frac{\mathring{\beta}_{d,d'}}{\mathring{\beta}_d}.$$

- The stationary probability of being in state $d \in \mathcal{D}$ is

$$\mathring{\pi}_d = \frac{\mathring{\beta}_d}{\sum_{\bar{d}\in\mathcal{D}}\mathring{\beta}_{\bar{d}}}.$$

This time-invariant Markov process can be interpreted as a backtrackless random walk on the normal graph N (or the normal graph $\mathring{\mathsf{N}}$), in the following called the $\mathring{\boldsymbol{\beta}}$-induced random walk on N (or $\mathring{\mathsf{N}}$). □

Some comments about the Markov process / random walk in Definition 13 are in order.

- If the Markov process is indecomposable and aperiodic, then the above stationary distribution is unique. Otherwise, there are multiple stationary distributions, and the one given above is just one possible stationary distribution.
- With the help of Lemma 12, it can easily be verified that $\{\mathring{\pi}_d\}_{d\in\mathcal{D}}$ is indeed a valid stationary distribution and that $\{\mathring{p}_{d,d'}\}_{d\in\mathcal{D},d'\in\mathcal{D}}$ are indeed valid transition probabilities. In fact, defining the vector $\mathring{\boldsymbol{\pi}} \triangleq (\mathring{\pi})_{d\in\mathcal{D}}$ and the matrix $\mathring{\boldsymbol{P}} \triangleq (\mathring{p}_{d,d'})_{d\in\mathcal{D},d'\in\mathcal{D}}$, we can write
$$\mathring{\boldsymbol{\pi}} = \mathring{\boldsymbol{\pi}} \cdot \mathring{\boldsymbol{P}}, \tag{4}$$
i.e., $\mathring{\boldsymbol{\pi}}$ is "self-consistent" according to the definition used in the introductory section.
- Let $\boldsymbol{\epsilon} \triangleq \boldsymbol{\psi}_{\text{edge}}(\mathring{\boldsymbol{\beta}})$. From Definitions 11 and 13 it follows that there is a $\mathring{\boldsymbol{\beta}}$-dependent constant $\gamma \in \mathbb{R}_{>0}$ such that
$$\sum_{d\in\mathcal{D}_e} \mathring{\pi}_d = \gamma \cdot \epsilon_e$$
for all $e \in \mathcal{E}$. This observation implies that the probability for the random walk to visit edge $e \in \mathcal{E}$ is proportional to the corresponding coordinate of the edge-based pseudo-codeword $\boldsymbol{\epsilon}$.
- A Markov chain is called time-reversible [26, Chapter 4.3] if the probability of visiting a sequence of states is unchanged when reversing the order of the states in the sequence. It can be verified that the Markov chain / random walk at hand is time-reversible in the following generalized sense. For any $T \in \mathbb{Z}_{>0}$, let $\boldsymbol{d}' = (d_1,\ldots,d_T) \in \mathcal{D}^T$ and $\boldsymbol{d} = (d_1',\ldots,d_T') \in \mathcal{D}^T$ be two sequences of directed edges such that $e(d_t) = e(d_t')$ and $d_t \neq d_t'$ for all $t \in \{1,\ldots,T\}$. Then
$$\mathring{\pi}_{d_1}\cdot\mathring{p}_{d_1,d_2}\cdots\mathring{p}_{d_{T-1},d_T} = \mathring{\pi}_{d_T'}\cdot\mathring{p}_{d_T',d_{T-1}'}\cdots\mathring{p}_{d_2',d_1'}.$$
This property of the Markov chain / random walk was implicitly a key part of the proofs in [10].
- For any $\boldsymbol{\beta} \in \mathcal{B}$ to which a $\mathring{\boldsymbol{\beta}} \in \mathring{\mathcal{B}}$ can be associated according to Definition 9, we will call the $\mathring{\boldsymbol{\beta}}$-induced random walk also the $\boldsymbol{\beta}$-induced random walk.

V. Convex Cones that are Supersets of the Edge-Based Fundamental Cone

This section features Theorem 21, the main result of the present paper. This theorem shows that certain convex cones are supersets of the edge-based fundamental cone. We will assume that the normal graph N is connected.

Definition 14: For a normal graph N, the graph distance $\Delta_{\mathsf{N}}(f,f') \in \mathbb{Z}_{\geqslant 0}$ between the two function nodes $f, f' \in \mathcal{F}$ is defined as the length of the shortest path that connects f with

f'. The graph distance $\Delta_{\mathsf{N}}(f,e) \in \mathbb{Z}_{\geqslant 0}$ between a function node $f \in \mathcal{F}$ and an edge $e \in \mathcal{E}$ is defined to be t if the shortest path connecting f with e is $f{=}f_0, e_0, f_1, e_1, \ldots, f_t, e_t{=}e$.

The girth N of a normal graph N is defined to be the length of the shortest cycle in N. Throughout this section, we fix some scalar $T \in \mathbb{Z}_{\geqslant 0}$ with $T \leqslant \frac{1}{2}\operatorname{girth}(\mathsf{N}) - 1$ and define $\mathcal{T} \triangleq \{0, 1, \ldots, T\}$. Moreover, we fix some vector $\boldsymbol{\xi} \in \mathbb{R}_{\geqslant 0}^{T+1}$. For $t > T$, we define $\xi_t \triangleq 0$.

Remark 15: Although the definitions, statements, and proofs in this section will assume that T and N are such that $T \leqslant \frac{1}{2}\operatorname{girth}(\mathsf{N}) - 1$, i.e., that T is bounded from above for a given normal graph N, this constraint can easily be removed as we will discuss in Remark 22 at the end of this section.

The following definition is motivated by the concept of valid deviations in computation trees [12], [10].

Definition 16: For every $f_0 \in \mathcal{F}$, define the normal graph $\hat{\mathsf{N}}^{(f_0)}\big(\hat{\mathcal{F}}^{(f_0)}, \hat{\mathcal{E}}^{(f_0)}, \hat{\mathcal{A}}^{(f_0)}, \hat{\mathcal{G}}^{(f_0)}\big)$ as follows.

- The factor node set $\hat{\mathcal{F}}^{(f_0)}$ contains all function nodes $f \in \mathcal{F}$ such that $\Delta_{\mathsf{N}}(f_0, f) \leqslant T$.
- The function node $f_0 \in \hat{\mathcal{F}}^{(f_0)}$ will be called the root node of $\hat{\mathsf{N}}^{(f_0)}$.
- The edge set $\hat{\mathcal{E}}^{(f_0)}$ contains all the edges $e \in \mathcal{E}$ such that $\Delta_{\mathsf{N}}(f_0, e) \leqslant T$.
- Based on $\hat{\mathcal{E}}^{(f_0)}$, the set $\hat{\mathcal{D}}^{(f_0)}$ is defined analogously to the way that the set $\mathcal{D}$ is defined based on $\mathcal{E}$.
- For any function node $f \in \hat{\mathcal{F}}^{(f_0)}$ and any edge $e \in \hat{\mathcal{E}}_f$, the edge e will be called inward with respect to f if e lies on a path from f to f_0. Otherwise e will be called outward with respect to f.
- For any function node $f \in \hat{\mathcal{F}}^{(f_0)}$ and any directed edge $d \in \hat{\mathcal{D}}_f$, the directed edge d will be called inbound with respect to f if d lies on a directed path from f to f_0. Otherwise d will be called outbound with respect to f.
- For $f = f_0$ we define $\hat{\mathcal{A}}_f^{(f_0)} \triangleq \mathcal{A}_f \setminus \{\mathbf{0}\}$.
- For every $f \in \hat{\mathcal{F}}^{(f_0)} \setminus \{f_0\}$ we define

$$\hat{\mathcal{A}}_f^{(f_0)} \triangleq \big\{\hat{\boldsymbol{a}}_f \in \mathcal{A}_f \;\big|\; \hat{\boldsymbol{a}}_f = \mathbf{0} \text{ or } \hat{a}_{f,e} = 1\big\},$$

  where $e \in \hat{\mathcal{E}}_f^{(f_0)}$ is inward with respect to f.
- For every $e \in \hat{\mathcal{E}}^{(f_0)}$ we define $\hat{\mathcal{A}}_e^{(f_0)} \triangleq \mathcal{A}_e$.
- The local functions $\hat{g}_f$, $f \in \mathcal{F}$, are left unspecified, but their respective supports are assumed to match the sets $\hat{\mathcal{A}}_f$, $f \in \mathcal{F}$. □

In the same way as $\mathcal{C}_{\text{edge}}$ is defined based on N, we will define, for every $f_0 \in \mathcal{F}$, the set of valid configurations $\hat{\mathcal{C}}_{\text{edge}}^{(f_0)}$ of $\hat{\mathsf{N}}$. Note that $\hat{\mathcal{C}}_{\text{edge}}^{(f_0)}$ contains all (minimal *and* non-minimal) valid deviations of the computation tree rooted at f_0 and of depth T.

Example 17: Consider a parity-check matrix $\boldsymbol{H}$ of some $(3,4)$-regular LDPC code[5] and associate with it a normal graph N as defined in Example 3. If $\operatorname{girth}(\mathsf{N}) \geqslant 10$ then we can choose $T = 4$. Figure 3 shows the resulting normal graph $\hat{\mathsf{N}}^{(f_0)}$ when f_0 is chosen to be one of the equal function nodes. The normal graph N has the following alphabets.

- $\mathcal{A}_f = \{(0,0,0),\ (1,1,1)\}$ if $f \in \mathcal{I}$.
- $\mathcal{A}_f$ contains all eight binary length-4 vectors with even Hamming weight if $f \in \mathcal{J}$.
- $\mathcal{A}_e = \{0,1\}$ for all $e \in \mathcal{E}$.

On the other hand, the normal graph $\hat{\mathsf{N}}^{(f_0)}$, $f_0 \in \mathcal{F}$, has the following alphabets.

- $\hat{\mathcal{A}}_f = \{(1,1,1)\}$ if $f = f_0 \in \mathcal{I}$.
- $\hat{\mathcal{A}}_f = \{(0,0,0),\ (1,1,1)\}$ if $f \in \big(\hat{\mathcal{F}}^{(f_0)} \setminus \{f_0\}\big) \cap \mathcal{I}$.
- $\hat{\mathcal{A}}_f$ contains all eight binary length-4 vectors with even Hamming weight if $f = f_0 \in \mathcal{J}$.
- $\hat{\mathcal{A}}_f = \{(0,0,0,0),\ (1,1,0,0),\ (1,0,1,0),\ (1,0,0,1),\ (1,1,1,1)\}$ when $f \in \big(\hat{\mathcal{F}}^{(f_0)} \setminus \{f_0\}\big) \cap \mathcal{J}$. (Here the vectors are ordered such that the first component corresponds to the inward edge with respect to f.)
- $\hat{\mathcal{A}}_e = \{0,1\}$ for all $e \in \hat{\mathcal{E}}^{(f_0)}$.

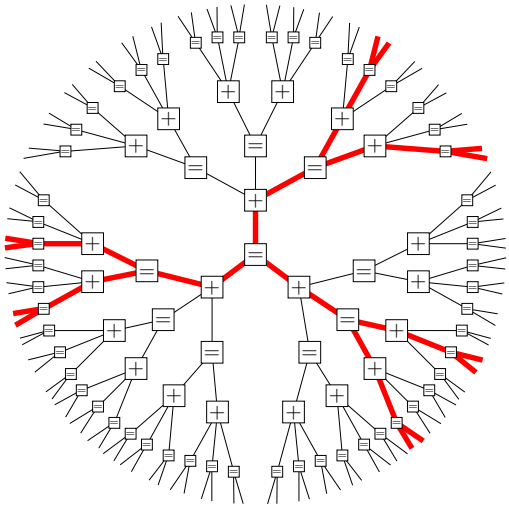

Fig. 3. Normal graph $\hat{\mathsf{N}}^{(f_0)}$ for Example 17 where the root function node f_0 is chosen to be one of the equal function nodes in N and where $T = 4$. The thick red edges highlight the non-zero part of some valid deviation, see the text at the end of Example 17.

Figure 3 shows a possible valid deviation $\hat{\boldsymbol{c}} \in \hat{\mathcal{C}}_{\text{edge}}^{(f_0)}$, where a thick red edge $e \in \hat{\mathcal{E}}^{(f_0)}$ corresponds to $\hat{c}_e = 1$ and where a thin black edge $e \in \hat{\mathcal{E}}^{(f_0)}$ corresponds to $\hat{c}_e = 0$. Observe that $\hat{\boldsymbol{c}}$ happens to be a *minimal* valid deviation. □

Note that the little hat on top of $\hat{\mathsf{N}}$, etc., is mnemonic for the fact that the non-zero part of a valid configuration in $\hat{\mathsf{N}}$ always looks similar to the thick-red-edge-subgraph in Figure 3, i.e., it is a tree rooted at f_0.

The following definition introduces some graph-dependent weighting factors. These are crucial for extending the results in [10] to normal graphs beyond normal graphs of $(w_{\text{col}}, w_{\text{row}})$-regular LDPC codes.

Definition 18: For every $f_0 \in \mathcal{F}$ and for any $\hat{\boldsymbol{c}} \in \hat{\mathcal{C}}_{\text{edge}}^{(f_0)}$, the vector $\boldsymbol{\chi}^{(\hat{\boldsymbol{c}})}$ is defined as follows.[6] If $e \in \mathcal{E} \setminus \hat{\mathcal{E}}^{(f_0)}$

[5] A code is called a $(w_{\text{col}}, w_{\text{row}})$-regular LDPC code if it is defined by a parity-check matrix with uniform column weight w_{col} and uniform row weight w_{row}.

[6] Possibly more precise would be $\hat{\boldsymbol{c}}^{(f_0)}$ instead of $\hat{\boldsymbol{c}}$, but we will prefer the more concise latter notation.

then $\chi_e^{(\hat{\boldsymbol{c}})} \triangleq 0$. Otherwise, let $t \triangleq \Delta_{\mathsf{N}}(f_0, e)$, and let $f_0, e_0, f_1, e_1, \ldots, f_t, e_t{=}e$ be the shortest path from f_0 to e in N. Then[7]

$$\chi_e^{(\hat{\boldsymbol{c}})} \triangleq \prod_{s=1}^{t} \frac{1}{w_{\mathrm{H}}(\hat{\boldsymbol{c}}_{f_s}) - 1}. \tag{5}$$

□

Definition 19: For every $f_0 \in \mathcal{F}$ and for every $\hat{\boldsymbol{c}} \in \hat{\mathcal{C}}_{\mathrm{edge}}^{(f_0)}$ define the vector $\boldsymbol{\epsilon}^{(\hat{\boldsymbol{c}})} \in \mathbb{R}_{\geqslant 0}^{|\mathcal{E}|}$ with components

$$\epsilon_e^{(\hat{\boldsymbol{c}})} \triangleq \hat{c}_e^{(f_0)} \cdot \chi_e^{(\hat{\boldsymbol{c}})} \cdot \xi_{\Delta_{\mathsf{N}}(f_0,e)}, \quad e \in \mathcal{E}.$$

□

Definition 20: For every $f_0 \in \mathcal{F}$, define $\mathcal{S}_{\mathrm{edge}}^{(f_0)}$ to be the set

$$\mathcal{S}_{\mathrm{edge}}^{(f_0)} \triangleq \left\{ \boldsymbol{\epsilon}^{(\hat{\boldsymbol{c}})} \;\middle|\; \hat{\boldsymbol{c}} \in \hat{\mathcal{C}}_{\mathrm{edge}}^{(f_0)} \right\},$$

With this, we define $\mathcal{S}_{\mathrm{edge}}$ to be the set

$$\mathcal{S}_{\mathrm{edge}} \triangleq \bigcup_{f_0 \in \mathcal{F}} \mathcal{S}_{\mathrm{edge}}^{(f_0)}.$$

Moreover, we let $\overline{\mathcal{K}}_{\mathrm{edge}}$ be the conic hull of $\mathcal{S}_{\mathrm{edge}}$, i.e.,

$$\overline{\mathcal{K}}_{\mathrm{edge}} \triangleq \mathrm{conic}\left(\mathcal{S}_{\mathrm{edge}}\right),$$

i.e., the vectors in $\mathcal{S}_{\mathrm{edge}}$ "span" the convex cone $\overline{\mathcal{K}}_{\mathrm{edge}}$. □

Theorem 21: With the assumptions on T, $\mathcal{T}$, and $\boldsymbol{\xi}$ made at the beginning of this section, in particular also Remark 15, and with Definitions 16, 18, 19, and 20, it holds that

$$\mathcal{K}_{\mathrm{edge}} \subseteq \overline{\mathcal{K}}_{\mathrm{edge}}.$$

Proof: (Sketch.) Choose any $\boldsymbol{\epsilon}$ in $\mathcal{K}_{\mathrm{edge}} \cap \mathcal{O}$. (Any $\boldsymbol{\epsilon}$ in $\mathcal{K}_{\mathrm{edge}}$ can always be rescaled by a positive scalar such that $\boldsymbol{\epsilon}$ is in $\mathcal{K}_{\mathrm{edge}} \cap \mathcal{O}$.) We have to show that $\boldsymbol{\epsilon}$ is in $\overline{\mathcal{K}}_{\mathrm{edge}}$, which is equivalent to showing that $\boldsymbol{\epsilon}$ is in the conic hull of $\mathcal{S}_{\mathrm{edge}}$, which is equivalent to showing that $2 \cdot \left(\sum_{t \in \mathcal{T}} \xi_t\right) \cdot \boldsymbol{\epsilon}$ is in the conic hull of $\mathcal{S}_{\mathrm{edge}}$, which is equivalent to showing that

$$2 \cdot \left(\sum_{t \in \mathcal{T}} \xi_t\right) \cdot \boldsymbol{\epsilon} = \sum_{f_0 \in \mathcal{F}} \sum_{\hat{\boldsymbol{c}} \in \hat{\mathcal{C}}_{\mathrm{edge}}^{(f_0)}} \gamma_{f_0,\hat{\boldsymbol{c}}} \cdot \boldsymbol{\epsilon}^{(\hat{\boldsymbol{c}})}$$

for some suitable non-negative constants $\{\gamma_{f_0,\hat{\boldsymbol{c}}}\}_{f_0,\hat{\boldsymbol{c}}}$. This conclusion can indeed be established by choosing the constants

$$\gamma_{f_0,\hat{\boldsymbol{c}}} \triangleq \beta_{f_0,\hat{\boldsymbol{c}}_{f_0}} \cdot \prod_{f \in \hat{\mathcal{F}}^{(f_0)}} \left(\frac{\beta_{f,\hat{\boldsymbol{c}}_f}}{\beta_{e(f_0,f),1}}\right)^{[\hat{\boldsymbol{c}}_f \neq \mathbf{0}]},$$

$f_0 \in \mathcal{F}$, $\hat{\boldsymbol{c}} \in \hat{\mathcal{C}}_{\mathrm{edge}}^{(f_0)}$, where $\boldsymbol{\beta}$ is chosen such that $\boldsymbol{\psi}_{\mathrm{edge}}(\boldsymbol{\beta}) = \boldsymbol{\epsilon}$, and where $e(f_0, f)$ denotes the inward edge with respect to f in $\hat{\mathsf{N}}^{(f_0)}$. Without going into the details, let us mention that the above summation over $\hat{\boldsymbol{c}} \in \hat{\mathcal{C}}_{\mathrm{edge}}^{(f_0)}$ is carried out by defining a suitable cycle-free normal graph and by applying the sum-product algorithm. Finally, the summation over $f_0 \in \mathcal{F}$ is carried out by taking advantage of the properties of the $\boldsymbol{\beta}$-induced random walk □

[7]Note that the product in (5) starts at $s = 1$. Redefining the product to start at $s = 0$ is also possible, and leads to a rescaling of the vectors $\boldsymbol{\epsilon}^{(\hat{\boldsymbol{c}})}$ in the upcoming Definition 19, but it does not change the statement in the upcoming Theorem 21.

Let us comment on this result.

- The set $\mathcal{S}_{\mathrm{edge}}$ contains vectors that are defined based on minimal local deviations in computation trees of depth T and rooted at all possible function nodes $f_0 \in \mathcal{F}$. Two types of weighting constants appear in the construction of the elements of $\mathcal{S}_{\mathrm{edge}}$. First, the non-negative weighting vector $\boldsymbol{\xi}$ can be chosen freely. Secondly, the non-negative weighting vectors $\left(\boldsymbol{\chi}^{(\hat{\boldsymbol{c}})}\right)_{f_0,\hat{\boldsymbol{c}}}$ are a function of $\hat{\boldsymbol{c}}$, and therefore implicitly also a function of the structure of the normal graph N.
- Because of the way that the weighting vectors $\left(\boldsymbol{\chi}^{(\hat{\boldsymbol{c}})}\right)_{f_0,\hat{\boldsymbol{c}}}$ were defined in (5), one can show that for any *non-minimal* valid deviation $\hat{\boldsymbol{c}} \in \hat{\mathcal{C}}_{\mathrm{edge}}^{(f_0)}$ the vector $\boldsymbol{\epsilon}^{(\hat{\boldsymbol{c}})}$ can be written as a conic combination of vectors in $\mathcal{S}_{\mathrm{edge}}^{(f_0)}$ that correspond to *minimal* valid deviations in $\hat{\mathcal{C}}_{\mathrm{edge}}^{(f_0)}$. Therefore, for the purpose of defining $\overline{\mathcal{K}}_{\mathrm{edge}}$, it is sufficient to include only the minimal valid deviations $\hat{\boldsymbol{c}} \in \hat{\mathcal{C}}_{\mathrm{edge}}^{(f_0)}$ in the definition of $\mathcal{S}_{\mathrm{edge}}^{(f_0)}$ (cf. skinny trees in [10]).
- The importance of the random walks for Theorem 21, especially the "self-consistency property" of the corresponding stationary distribution vector, can be seen as follows. Using the same notation as in the proof sketch of Theorem 21, let $\mathring{\boldsymbol{\pi}}$ be the stationary distribution vector of the $\boldsymbol{\beta}$-induced random walk. Moreover, for every $f_0 \in \mathcal{F}$ we define the vector $\mathring{\boldsymbol{\pi}}^{(f_0)} \in \mathbb{R}^{|\mathcal{D}|}$ with components $\mathring{\pi}_d^{(f_0)} = \pi_d \cdot \left[d \in \mathcal{D}_{f_0}\right]$, $d \in \mathcal{D}$. Then

$$\begin{aligned}\boldsymbol{\delta} \triangleq \left(\sum_{t \in \mathcal{T}} \xi_t\right) \cdot \mathring{\boldsymbol{\pi}} &= \sum_{t \in \mathcal{T}} \xi_t \cdot \mathring{\boldsymbol{\pi}} \overset{\text{(a)}}{=} \sum_{t \in \mathcal{T}} \xi_t \cdot \left(\mathring{\boldsymbol{\pi}} \cdot \mathring{\boldsymbol{P}}^t\right) \\ &\overset{\text{(b)}}{=} \sum_{t \in \mathcal{T}} \xi_t \cdot \sum_{f_0 \in \mathcal{F}} \mathring{\boldsymbol{\pi}}^{(f_0)} \cdot \mathring{\boldsymbol{P}}^t = \sum_{f_0 \in \mathcal{F}} \underbrace{\mathring{\boldsymbol{\pi}}^{(f_0)} \cdot \sum_{t \in \mathcal{T}} \xi_t \cdot \mathring{\boldsymbol{P}}^t}_{\triangleq \boldsymbol{\delta}^{(f_0)}},\end{aligned} \tag{6}$$

  where at step (a) we have used the "self-consistency property" (4) multiple times, and where at step (b) we have used $\mathring{\boldsymbol{\pi}} = \sum_{f_0 \in \mathcal{F}} \mathring{\boldsymbol{\pi}}^{(f_0)}$.
  Fix some $e \in \mathcal{E}$. Because of the way that $\boldsymbol{\delta}$ and $\mathring{\boldsymbol{\pi}}$ are defined, we observe that $\sum_{d \in \mathcal{D}_e} \delta_d = \sum_{d \in \mathcal{D}_e} \left(\sum_{t \in \mathcal{T}} \xi_t\right) \cdot \mathring{\pi}_d = \gamma \cdot \left(\sum_{t \in \mathcal{T}} \xi_t\right) \cdot \epsilon_e$, for some $\gamma \in \mathbb{R}_{>0}$. On the other hand, $\sum_{d \in \mathcal{D}_e} \delta_d^{(f_0)}$ is only non-zero for $e \in \hat{\mathcal{E}}^{(f_0)}$, i.e., it is only non-zero on edges that belong to the local deviations normal graph $\hat{\mathsf{N}}^{(f_0)}$. Therefore, (6) shows how an edge-based pseudo-codeword can be written as a conic combination of vectors that are non-zero only on the edges defined by computation trees.
  Of course, this is not a proof of Theorem 21 (in particular, $\boldsymbol{\delta}^{(f_0)}$ needs to be related to valid deviations in $\hat{\mathsf{N}}^{(f_0)}$) but it goes a long way towards obtaining a proof and gaining some intuition about it.
  On the side we note that for minimal pseudo-codewords we have $\mathring{\boldsymbol{\pi}}^{(f_0)} \cdot \sum_{t=T'}^{T'+T''} \mathring{\boldsymbol{P}}^t \overset{T' \to \infty}{\longrightarrow} \gamma \cdot \mathring{\boldsymbol{\pi}}$, for suitable $T'' \in \mathbb{Z}_{>0}$ and $\gamma \in \mathbb{R}_{>0}$.
- For a binary linear code $\mathcal{C}$ defined by some parity-check

matrix $\boldsymbol{H}$, along with the normal graph as defined in Example 3, one can easily derive a convex cone $\overline{\mathcal{K}}$ based on $\overline{\mathcal{K}}_{\mathrm{edge}}$ such that $\mathcal{K} \subseteq \overline{\mathcal{K}}$ as in Figure 1.

- The definitions and proofs in this section can be extended such that for *bipartite* normal graphs with function node classes $\mathcal{I}$ and $\mathcal{J}$ one can define a weighting vector $\boldsymbol{\xi}^{(\mathcal{I})}$ for all $f_0 \in \mathcal{I}$, and a weighting vector $\boldsymbol{\xi}^{(\mathcal{J})}$ for all $f_0 \in \mathcal{J}$.
- For a $(w_{\mathrm{col}}, w_{\mathrm{row}})$-regular LDPC codes, the weighting vectors $\boldsymbol{\xi}^{(\mathcal{I})}$ and $\boldsymbol{\xi}^{(\mathcal{J})}$ can be chosen so as to reconstruct the results of [10] for any choice of $(w_1, \ldots, w_{T'})$ in [10]. In particular, choosing $T \triangleq 2T'$, $\xi_0^{(\mathcal{I})} = 0$, $\xi_1^{(\mathcal{I})} = w_1$, $\xi_2^{(\mathcal{I})} = w_1(w_{\mathrm{col}} - 1)$, $\xi_3^{(\mathcal{I})} = w_2(w_{\mathrm{col}} - 1)$, $\xi_4^{(\mathcal{I})} = w_2(w_{\mathrm{col}} - 1)^2$, $\ldots$, $\xi_{T-1}^{(\mathcal{I})} = w_{T'}(w_{\mathrm{col}} - 1)^{T'-2}$, $\xi_T^{(\mathcal{I})} = w_{T'}(w_{\mathrm{col}} - 1)^{T'-1}$, and $\boldsymbol{\xi}^{(\mathcal{J})} = \mathbf{0}$, gives the connection.

Remark 22: As mentioned at the beginning of this section, all definitions, theorems, and proofs in this section can easily be extended to the case where T is chosen independently of the girth of N. This is accomplished as follows. Namely, for any given T there is an $M \in \mathbb{Z}_{>0}$ such that there is an M-fold graph cover $\tilde{\mathsf{N}}$ [3], [4] of the base graph N such that $T \leqslant \frac{1}{2}\mathrm{girth}(\tilde{\mathsf{N}}) - 1$. Choose such a graph cover and then apply all the definitions of this section to this graph cover. Finally, the vectors of the set $\mathcal{S}_{\mathrm{edge}}$ are obtained by projecting down the vectors of the set $\tilde{\mathcal{S}}_{\mathrm{edge}}$. Specifically, $\boldsymbol{\epsilon} \in \mathcal{S}_{\mathrm{edge}}$ is obtained from $\tilde{\boldsymbol{\epsilon}} \in \tilde{\mathcal{S}}_{\mathrm{edge}}$ via $\epsilon_e = \frac{1}{M}\sum_{m=1}^{M} \tilde{\epsilon}_{(e,m)}$, $e \in \mathcal{E}$.

VI. Connections Between the Bethe Entropy and the Random Walk Entropy Rate

Omitted.

VII. Conclusions

In this paper we have generalized the geometrical aspects of the paper [10]. In particular, we have seen that the girth of the normal graph does not impose restrictions on the above construction of supersets of the fundamental cone.

An interesting avenue for further research is to see how the LP decoding performance guarantees that are obtained in the density evolution part of the paper [10] can be modified so as to put less restrictions on the girth of the normal graph.

Given the connection that was established in [27], [28] between compressed sensing LP decoding [29] and channel coding LP decoding [1], [2], it will be interesting to see what implications the techniques in this paper have on compressed sensing LP decoding.

References

[1] J. Feldman, "Decoding error-correcting codes via linear programming," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, 2003.

[2] J. Feldman, M. J. Wainwright, and D. R. Karger, "Using linear programming to decode binary linear codes," *IEEE Trans. Inf. Theory*, vol. 51, no. 3, pp. 954–972, Mar. 2005.

[3] R. Koetter and P. O. Vontobel, "Graph covers and iterative decoding of finite-length codes," in *Proc. 3rd Intern. Symp. on Turbo Codes and Related Topics*, Brest, France, Sept. 1–5 2003, pp. 75–82.

[4] P. O. Vontobel and R. Koetter, "Graph-cover decoding and finite-length analysis of message-passing iterative decoding of LDPC codes," *accepted for IEEE Trans. Inform. Theory, available online under* `http://www.arxiv.org/abs/cs.IT/0512078`, 2007.

[5] ——, "Bounds on the threshold of linear programming decoding," in *Proc. IEEE Information Theory Workshop*, Punta Del Este, Uruguay, Mar. 13–16 2006, pp. 175–179.

[6] J. Feldman, T. Malkin, R. A. Servedio, C. Stein, and M. J. Wainwright, "LP decoding corrects a constant fraction of errors," *IEEE Trans. Inf. Theory*, vol. 53, no. 1, pp. 82–89, Jan. 2007.

[7] C. Daskalakis, A. G. Dimakis, R. M. Karp, and M. J. Wainwright, "Probabilistic analysis of linear programming decoding," *IEEE Trans. Inf. Theory*, vol. 54, no. 8, pp. 3565–3578, Aug. 2008.

[8] R. M. Tanner, "A recursive approach to low-complexity codes," *IEEE Trans. Inf. Theory*, vol. 27, no. 5, pp. 533–547, Sept. 1981.

[9] R. Koetter and P. O. Vontobel, "On the block error probability of LP decoding of LDPC codes," in *Proc. Inaugural Workshop of the Center for Information Theory and Applications*, UC San Diego, La Jolla, CA, USA, Feb. 6–10 2006, available online under `http://www.arxiv.org/abs/cs.IT/0602086`.

[10] S. Arora, C. Daskalakis, and D. Steurer, "Message-passing algorithms and improved LP decoding," in *Proc. 41st Annual ACM Symp. Theory of Computing*, Bethesda, MD, USA, May 31–June 2 2009.

[11] P. O. Vontobel and R. Koetter, "On the relationship between linear programming decoding and min-sum algorithm decoding," in *Proc. Intern. Symp. on Inf. Theory and its Applications (ISITA)*, Parma, Italy, Oct. 10–13 2004, pp. 991–996.

[12] N. Wiberg, "Codes and decoding on general graphs," Ph.D. dissertation, Linköping University, Sweden, 1996.

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

[14] G. D. Forney, Jr., "Codes on graphs: normal realizations," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 520–548, Feb. 2001.

[15] H.-A. Loeliger, "An introduction to factor graphs," *IEEE Sig. Proc. Mag.*, vol. 21, no. 1, pp. 28–41, Jan. 2004.

[16] N. Halabi and G. Even, "LP decoding of regular LDPC codes in memoryless channels," *in preparation*, Jan. 2010.

[17] T. Richardson and R. Urbanke, *Modern Coding Theory*. New York, NY: Cambridge University Press, 2008.

[18] R. G. Gallager, *Low-Density Parity-Check Codes*. M.I.T. Press, Cambridge, MA, 1963.

[19] P. O. Vontobel, "Connecting the Bethe entropy and the edge zeta function of a cycle code," *submitted to 2010 IEEE International Symposium on Information Theory*, Jan. 2010.

[20] R. Koetter, W.-C. W. Li, P. O. Vontobel, and J. L. Walker, "Characterizations of pseudo-codewords of (low-density) parity-check codes," *Adv. in Math.*, vol. 213, no. 1, pp. 205–229, Aug. 2007.

[21] P. O. Vontobel, "A factor-graph-based random walk, and its relevance for LP decoding analysis and Bethe entropy characterization," *in preparation, to be submitted to IEEE Trans. Inf. Theory*, Jan. 2010.

[22] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, UK: Cambridge University Press, 2004.

[23] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Constructing free-energy approximations and generalized belief propagation algorithms," *IEEE Trans. Inf. Theory*, vol. 51, no. 7, pp. 2282–2312, July 2005.

[24] M. J. Wainwright and M. I. Jordan, *Graphical models, exponential families, and variational inference*. Technical Report No. 649, Dept. of Statistics, UC Berkeley, Berkeley, CA, USA, 2003.

[25] H. M. Stark and A. A. Terras, "Zeta functions of finite graphs and coverings," *Adv. Math.*, vol. 121, no. 1, pp. 124–165, 1996.

[26] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, ser. Wiley Series in Telecommunications. New York: John Wiley & Sons Inc., 1991, a Wiley-Interscience Publication.

[27] A. G. Dimakis and P. O. Vontobel, "LP decoding meets LP decoding: a connection between channel coding and compressed sensing," in *Proc. 47th Allerton Conf. on Communications, Control, and Computing*, Allerton House, Monticello, Illinois, USA, Sep. 30–Oct. 2 2009.

[28] A. G. Dimakis, R. Smarandache, and P. O. Vontobel, "Channel coding LP decoding and compressed sensing LP decoding: further connections," in *Proc. 2010 Intern. Zurich Seminar on Communications*, Zurich, Switzerland, Mar. 3–5 2010.

[29] E. J. Candes and T. Tao, "Decoding by linear programming," *IEEE Trans. Inf. Theory*, vol. 51, no. 12, pp. 4203–4215, Dec. 2005.