Partition Functions of Normal Factor Graphs

G. David Forney, Jr.
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
Cambridge, MA 02139, USA
forneyd@comcast.net

Pascal O. Vontobel
Hewlett–Packard Laboratories
1501 Page Mill Road
Palo Alto, CA 94304, USA
pascal.vontobel@ieee.org

***Abstract*—One of the most common types of functions in mathematics, physics, and engineering is a sum of products, sometimes called a partition function. After "normalization," a sum of products has a natural graphical representation, called a normal factor graph (NFG), in which vertices represent factors, edges represent internal variables, and half-edges represent the external variables of the partition function. In physics, so-called trace diagrams share similar features.**

We believe that the conceptual framework of representing sums of products as partition functions of NFGs is an important and intuitive paradigm that, surprisingly, does not seem to have been introduced explicitly in the previous factor graph literature.

Of particular interest are NFG modifications that leave the partition function invariant. A simple subclass of such NFG modifications offers a unifying view of the Fourier transform, tree-based reparameterization, loop calculus, and the Legendre transform.

I. Introduction

Functions that can be expressed as sums of products are ubiquitous in mathematics, science, and engineering. Borrowing a physics term, we call such a function a *partition function*.

In this paper, we will represent partition functions by *normal factor graphs* (NFGs), which build on the concepts of factor graphs [14] and normal graphs [12]. A factor graph represents a product of factors by a bipartite graph, in which one set of vertices represents variables, while the other set of vertices represents factors. By introducing "normal" degree restrictions as in [12], we can represent a sum of products by an NFG in which edges represent variables and vertices represent factors. Moreover, internal and external variables are distinguished in an NFG by being represented by edges of degree 2 and degree 1, respectively. NFGs closely resemble the "Forney-style factor graphs" (FFGs) of Loeliger *et al.* [15], [16], with the difference that "closing the box" (summing over internal variables) is always explicitly assumed as part of the graph semantics.

There are as many applications of NFGs as there are of sums of products. In this paper, we will present several applications that highlight the usefulness of the graphical approach:

- **Trace diagrams**, which are closely related to NFGs, often provide insight into linear algebraic relations, particularly of the kind that arise in various areas of physics;
- The **sum-product algorithm** is naturally nicely derived in terms of NFGs;
- The **normal factor graph duality theorem** [2], [13] is a powerful general result, of which one corollary is the normal graph duality theorem of [12].
- The **holographic transformations** of NFGS of Al-Bashabsheh and Mao [2], which may be used to derive the "holographic algorithms" of Valiant [21] and others, may be further generalized to derive the "tree-based reparameterization" approach of Wainwright *et al.* [25], the "loop calculus" results of Chertkov and Chernyak [7], [8], and the Lagrange duality results of Vontobel and Loeliger [23], [24].
- **Linear codes defined on graphs** and their **weight generating functions** have natural representations as NFGs, as shown in [13], but we will not discuss this topic here.

II. Partition Functions and Graphs

A **partition function** is any function $Z(\mathbf{x})$ that is given in "sum-of-products form," as follows:

$$Z(\mathbf{x}) = \sum_{\mathbf{y}\in\mathcal{Y}} \prod_{k\in\mathcal{K}} f_k(\mathbf{x}_k, \mathbf{y}_k), \quad \mathbf{x} \in \mathcal{X},$$

where

- $\mathbf{X}$ is a set of m **external variables** X_i taking values x_i in alphabets $\mathcal{X}_i, 1 \le i \le m$;
- $\mathbf{Y}$ is a set of n **internal variables** Y_j taking values y_j in alphabets $\mathcal{Y}_i, 1 \le j \le n$;
- each **factor** $f_k(\mathbf{x}_k, \mathbf{y}_k), k \in \mathcal{K}$, is a function of certain subsets $\mathbf{X}_k \subseteq \mathbf{X}$ and $\mathbf{Y}_k \subseteq \mathbf{Y}$ of the sets of external and internal variables, respectively.

The set $\mathcal{X} = \prod_{i=1}^m \mathcal{X}_i$ of all possible external variable configurations is called the *domain* of the partition function, and the set $\mathcal{Y} = \prod_{j=1}^n \mathcal{Y}_j$ of all possible internal variable configurations is called its *configuration space*. We say that a factor $f_k(\mathbf{x}_k, \mathbf{y}_k)$ *involves* a variable X_i (resp. Y_j) if f_k is a function of that variable; *i.e.,* if $X_i \in \mathbf{X}_k$ (resp. $Y_j \in \mathbf{Y}_k$). For simplicity, we will assume that all functions are complex-valued, and that all variable alphabets are discrete.[1]

A particular sum-of-products form for a partition function will be called a *realization*. Different realizations that yield the same partition function $Z : \mathcal{X} \to \mathbb{C}$ will be called *equivalent*. We say that equivalent realizations *preserve the partition function*.

[1] Usually in physics a partition function is a sum over internal configurations (state configurations), and there are no external variables in our sense (although there may be parameters, such as temperature). So our usage of "partition function" extends the usual terminology of physics. Al-Bashabsheh and Mao [2] use the term "exterior function."

A. Normal partition functions

We will say that a realization of a partition function is **normal** if all external variables are involved in precisely one factor f_k, and all internal variables are involved in precisely two factors. These degree restrictions were introduced in [12] in the context of behavioral graphs.

As observed in [12], any realization may be converted to an equivalent normal realization by the following simple **normalization procedure**.

- For every external variable X_i, if X_i is involved in p factors, then define p *replica variables* $X_{i\ell}, 1 \leq \ell \leq p$, replace X_i by $X_{i\ell}$ in the ℓth factor in which X_i is involved, and introduce one new factor, namely an *equality indicator function* $\Phi_=(x_i, \{x_{i\ell}, 1 \leq \ell \leq p\})$ (see below).
- For every internal variable Y_j, if Y_j is involved in $q \geq 2$ factors, then define q replica variables $Y_{j\ell}, 1 \leq \ell \leq q$, replace Y_j by $Y_{j\ell}$ in the ℓth factor in which Y_j is involved, and introduce one new factor, namely an equality indicator function $\Phi_=(\{y_{j\ell}, 1 \leq \ell \leq q\})$.

Thus all replica variables are internal variables that are involved in precisely two factors, while the external variables X_i become involved in only one factor, namely an equality indicator function. Evidently this normalization procedure preserves the partition function.

B. Normal factor graphs

For a normal realization of a partition function, a natural graphical model is a **normal factor graph** (NFG), in which vertices are associated with factors, ordinary edges (*i.e.,* hyperedges of degree 2) are associated with internal variables, "half-edges" [12] (*i.e.,* hyperedges of degree 1) are associated with external variables, and a variable edge or half-edge is incident on a factor vertex if the variable is involved in that factor.

Example 1 (vector-matrix multiplication). Consider a multiplication $\mathbf{v} = \mathbf{w}M$ of a vector $\mathbf{w}$ by a matrix M, namely

$$v_j = \sum_{i \in \mathcal{I}} w_i M_{ij}, \quad j \in \mathcal{J},$$

for some discrete index sets $\mathcal{I}$ and $\mathcal{J}$. This may be interpreted as a normal realization of the function $v : \mathcal{J} \to \mathbb{C}$, with external variable J, internal variable I, and factors w_i and M_{ij}. Figure 1 shows the corresponding normal factor graph, in which the vertices are represented by labeled boxes, and the half-edge is represented by a special dongle symbol.[2] □

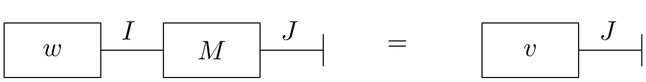

Fig. 1. Normal factor graph of a matrix multiplication $\mathbf{v} = \mathbf{w}M$.

[2]The dongle symbol "⊣" was chosen in [12] to suggest the possibility of a connection to another external half-edge in the manner of two railroad cars coupling, but of course this embellishment may be omitted.

C. Equality indicator functions

We use special symbols for certain frequently occurring factors. The most common and fundamental factor is the **equality indicator function** $\Phi_=$, which equals 1 if all incident variables (which must have a common alphabet) are equal, and equals 0 otherwise.

Figure 2 shows three ways of representing an equality indicator function: first, by a vertex labeled by $\Phi_=$; second, by a vertex labeled simply by an equality sign $=$; and third, as a junction vertex. The second representation makes a connection with the behavioral graph literature (*e.g.,* Tanner graphs), where vertices represent constraints rather than factors. The third representation makes connections with ordinary block diagrams, where any number of edges representing the same variable may meet at a junction, as well as with the factor graph literature, where variables are represented by vertices rather than by edges.

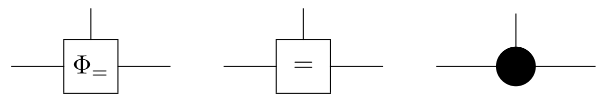

Fig. 2. Three representations of an equality indicator function of degree 3.

An equality indicator function of degree 2 is often denoted by a Kronecker delta function δ. Since such a function connects only two edges and constrains their respective variables to be equal, it may simply be omitted, as shown in Figure 3.[3]

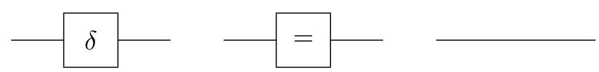

Fig. 3. Three representations of an equality indicator function of degree 2.

III. Trace Diagrams

It turns out that physicists have long used graphical diagrams called "trace diagrams" [10], [17], [18], [19], [20] that use semantics similar to those of NFGs. In this section we give a brief exposition of this topic, following [19].

In trace diagrams, the factors are often vectors, matrices, tensors, and so forth, and the variables are typically their indices. For instance, a matrix $M = \{M_{ij}, i \in \mathcal{I}, j \in \mathcal{J}\}$ may be considered to be a function of the two variables I and J, and is represented as a vertex with two incident edges, as in Figure 4(a).

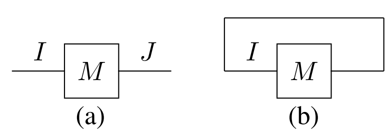

Fig. 4. Representations of (a) a matrix M; (b) the trace of M.

[3]The last equivalence shown in Figure 3 is actually a bit problematic, since a single edge is not a legitimate normal factor graph; however, as a component of a normal factor graph, such an edge is always incident on some factor vertex f_k, and since the combination of a factor f_k involving some internal variable Y_j with an equality function $\Phi_=(y_j, y_j')$ is just the same factor with Y_j' substituted for Y_j, this substitution can be made in any legitimate NFG (see also [2]).

Trace diagrams use the NFG convention that dangling edges (half-edges) represent external variables, whereas ordinary edges represent internal variables, and are to be summed over. For example, if the matrix M is square (*i.e.,* the index alphabets $\mathcal{I}$ and $\mathcal{J}$ are the same), and the half-edges representing I and J are connected as in Figure 4(b), then the resulting figure represents the *trace* of M, since $\mathrm{Tr}\ M = \sum_i M_{ii}$. This apparently explains why these kinds of graphical models are known as "trace diagrams."

The convention that indices that appear twice are implicitly to be summed over is known in physics as the *Einstein summation convention*. This convention is used rather generally in physics, not just with trace diagrams.

Trace diagrams permit visual proofs of various relationships in linear algebra. For example, Figure 5 proves the identity $\mathrm{Tr}\ ABC = \mathrm{Tr}\ BCA$.

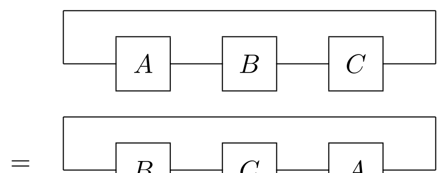

Fig. 5. Proof of the identity $\mathrm{Tr}\ ABC = \mathrm{Tr}\ BCA$.

If $\mathbf{u}$ and $\mathbf{v}$ are two real vectors with a common index set $\mathcal{I}$, then their *dot product* (inner product) is defined as

$$\mathbf{u} \cdot \mathbf{v} = \sum_{i \in \mathcal{I}} u_i v_i.$$

The trace diagram (or normal factor graph) of a dot product is illustrated in Figure 6(a).

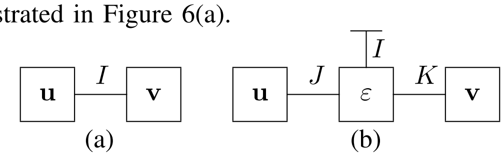

Fig. 6. Representations of (a) a dot product $\mathbf{u} \cdot \mathbf{v}$; (b) a cross product $\mathbf{u} \times \mathbf{v}$.

If $\mathbf{u}$ and $\mathbf{v}$ are two real three-dimensional vectors, then their *cross product* $\mathbf{u} \times \mathbf{v} = \mathbf{w}$ is defined by

$$\begin{aligned} w_1 &= u_2 v_3 - u_3 v_2; \\ w_2 &= u_3 v_1 - u_1 v_3; \\ w_3 &= u_1 v_2 - u_2 v_1. \end{aligned}$$

Equivalently,

$$w_i = \sum_{j=1}^{3} \sum_{k=1}^{3} \varepsilon_{ijk} u_j v_k,$$

where we use the *Levi-Civita symbol* ε_{ijk}, defined as

$$\varepsilon_{ijk} = \begin{cases} +1, & \text{if } ijk \text{ is an even permutation of 123;} \\ -1, & \text{if } ijk \text{ is an odd permutation of 123;} \\ 0, & \text{otherwise.} \end{cases}$$

Thus $\mathbf{w}$ is given in the form of a normal partition function with external variable I and internal variables J and K. The trace diagram or NFG of this cross product is illustrated in Figure 6(b). (Notice that in this case the order of the indices is important, since $\varepsilon_{ijk} = -\varepsilon_{jik}$.)

Similarly, the determinant of a 3×3 matrix M may be written in terms of ε_{ijk} as

$$\det M = \sum_{i=1}^{3} \sum_{j=1}^{3} \sum_{k=1}^{3} \varepsilon_{ijk} M_{1i} M_{2j} M_{3k}.$$

Thus if $\mathbf{M}_1, \mathbf{M}_2$ and $\mathbf{M}_3$ are the three rows of M, then its determinant may be represented in trace diagram or normal factor graph notation as in Figure 7.

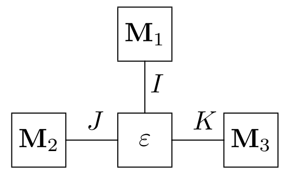

Fig. 7. Representation of a determinant $\det\{\mathbf{M}_1, \mathbf{M}_2, \mathbf{M}_3\}$.

Figure 7 shows that the determinant of M may be expressed in three equivalent ways, as follows:

$$\begin{aligned} \det M &= \mathbf{M}_1 \cdot (\mathbf{M}_2 \times \mathbf{M}_3) \\ &= \mathbf{M}_2 \cdot (\mathbf{M}_3 \times \mathbf{M}_1) \\ &= \mathbf{M}_3 \cdot (\mathbf{M}_1 \times \mathbf{M}_2). \end{aligned}$$

The trace diagram notation permits other operations that have not heretofore been considered in the factor graph literature. For example, two trace diagrams with the same sets of external variables that are connected by a plus or minus sign represent the *sum* or *difference* of the corresponding partition functions.[4] For example, Figure 8 illustrates the "contracted epsilon identity," namely

$$\sum_{k=1}^{3} \varepsilon_{ijk} \varepsilon_{k\ell m} = \delta_{i\ell}\delta_{jm} - \delta_{im}\delta_{j\ell}.$$

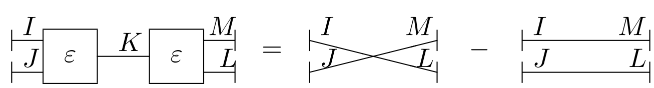

Fig. 8. Contracted epsilon identity.

From this identity, or its corresponding trace diagram, we can derive such identities as

$$(\mathbf{u} \times \mathbf{v}) \times \mathbf{w} = (\mathbf{u} \cdot \mathbf{w})\mathbf{v} - (\mathbf{v} \cdot \mathbf{w})\mathbf{u},$$

illustrated in Figure 9(a), or

$$(\mathbf{u} \times \mathbf{v}) \cdot (\mathbf{w} \times \mathbf{x}) = (\mathbf{u} \cdot \mathbf{w})(\mathbf{v} \cdot \mathbf{x}) - (\mathbf{u} \cdot \mathbf{x})(\mathbf{v} \cdot \mathbf{w}),$$

illustrated in Figure 9(b), which reduce expressions involving two cross products to simpler forms involving only dot products.

[4]A product of partition functions is represented simply by a disconnected factor graph, with each component graph representing a component function.

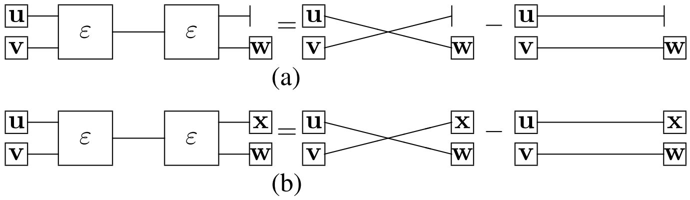

Fig. 9. Cross product identities: (a) $(\mathbf{u} \times \mathbf{v}) \times \mathbf{w} = (\mathbf{u} \cdot \mathbf{w})\mathbf{v} - (\mathbf{v} \cdot \mathbf{w})\mathbf{u}$; (b) $(\mathbf{u} \times \mathbf{v}) \cdot (\mathbf{w} \times \mathbf{x}) = (\mathbf{u} \cdot \mathbf{w})(\mathbf{v} \cdot \mathbf{x}) - (\mathbf{u} \cdot \mathbf{x})(\mathbf{v} \cdot \mathbf{w})$.

IV. THE SUM-PRODUCT ALGORITHM

The sum-product algorithm is an efficient method for computing partition functions of cycle-free graphs. It has been explained many times, including in [12]. Here we explain it again in the language of normal factor graphs, with the objective of achieving a clearer and more intuitive explanation than in [12]. We freely use ideas from *e.g.,* [1], [14], [15], [16], [26].

As Al-Bashabsheh and Mao [2] have emphasized, a partition function is completely determined by the set $\{f_k(\mathbf{x}_k, \mathbf{y}_k)\}$ of factors, independent of their ordering. In evaluating a partition function, factors may be arbitrarily ordered and grouped. This observation (called the "generalized distributive law" by Aji and McEliece [1]) is at the root of the sum-product algorithm.

We start with a normal realization of a partition function with *no external variables* whose associated normal graph $\mathcal{G}$ is *connected* and *cycle-free*. Thus the partition function of $\mathcal{G}$ is a constant, denoted by $Z(\mathcal{G})$, and $\mathcal{G}$ is an ordinary graph (no half-edges) that moreover is a tree.

A connected graph $\mathcal{G}$ is cycle-free if and only if any *cut* through any edge Y_j divides $\mathcal{G}$ into two disconnected graphs, which we label arbitrarily as $\overrightarrow{\mathcal{G}}_j$ and $\overleftarrow{\mathcal{G}}_j$. Such a cut divides the edge associated with Y_j into two half-edges associated with two external variables, denoted by $\overrightarrow{Y}_j$ and $\overleftarrow{Y}_j$, with the same alphabet $\mathcal{Y}_j$ as Y_j, as illustrated in Figure 10.

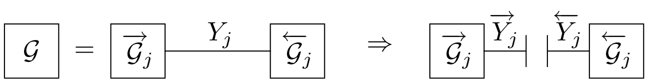

Fig. 10. Disconnecting a cycle-free NFG $\mathcal{G}$ by a cut through edge Y_j.

Let us define the *messages* $\overrightarrow{\mu}_j(y_j)$ and $\overleftarrow{\mu}_j(y_j)$ as the partition functions of $\overrightarrow{\mathcal{G}}_j$ and $\overleftarrow{\mathcal{G}}_j$, respectively; *i.e.,*

$$\overrightarrow{\mu}_j(y_j) = \sum_{\overrightarrow{\mathbf{y}} \in \overrightarrow{\mathbf{Y}}} \prod_{k \in \overrightarrow{\mathcal{K}}} f_k(\mathbf{y}_k),$$

where $\overrightarrow{\mathbf{Y}}$ is the set of left-side variables (excluding Y_j), and $\overrightarrow{\mathcal{K}}$ is the set of indices of left-side factors, and similarly for $\overleftarrow{\mu}_j(y_j)$. The goal of the sum-product algorithm is to compute the messages $\overrightarrow{\mu}_j(y_j), \overleftarrow{\mu}_j(y_j)$ for every internal variable Y_j.

To compute a message such as $\overrightarrow{\mu}_j(y_j)$, consider the factor vertex to which $\overrightarrow{Y}_j$ is attached. For simplicity, let us suppose that this vertex has degree 3, and that the associated factor is $f(y_j, y_{j'}, y_{j''})$, as shown in Figure 11.

Since $\mathcal{G}$ is cycle-free, the subgraphs $\overrightarrow{\mathcal{G}}_{j'}$ and $\overrightarrow{\mathcal{G}}_{j''}$ that extend from the edges $Y_{j'}$ and $Y_{j''}$ must be disjoint. Their partition functions, $\overrightarrow{\mu}_{j'}(y_{j'})$ and $\overrightarrow{\mu}_{j''}(y_{j''})$, include all factors

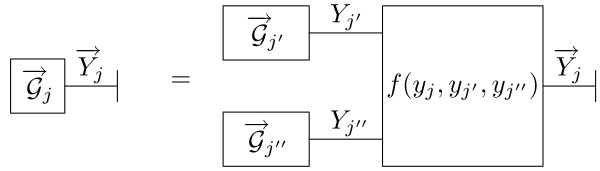

Fig. 11. Expressing an NFG in terms of subgraphs connected to a vertex.

in $\overrightarrow{\mu}_j(y_j)$ except $f(y_j, y_{j'}, y_{j''})$, and sum over all internal variables except $Y_{j'}$ and $Y_{j''}$. Therefore the partition function $\overrightarrow{\mu}_j(y_j)$ of $\overrightarrow{\mathcal{G}}_j$ may be expressed in terms of the partition functions of these subgraphs as follows:

$$\overrightarrow{\mu}_j(y_j) = \sum_{y_{j'} \in \mathcal{Y}_{j'}} \sum_{y_{j''} \in \mathcal{Y}_{j''}} f(y_j, y_{j'}, y_{j''}) \overrightarrow{\mu}_{j'}(y_{j'}) \overrightarrow{\mu}_{j''}(y_{j''}).$$

More generally, if the factor vertex to which edge Y_j is attached is $f_k(\mathbf{y}_k)$, then the message update rule is

$$\overrightarrow{\mu}_j(y_j) = \sum_{\mathbf{y}_k \setminus \{y_j\}} f_k(\mathbf{y}_k) \prod_{j' \in \mathcal{J}_k \setminus \{j\}} \overrightarrow{\mu}_{j'}(y_{j'}).$$

This is called the **sum-product update rule**.

Since $\mathcal{G}$ is connected and cycle-free, it is a tree (assuming that it is finite). Each message $\overrightarrow{\mu}_j$ has a *depth* equal to the maximum length of any path from that message to any leaf vertex. The messages at depth 1 can be computed immediately, the messages at depth 2 can be computed as soon as the messages at depth 1 are known, and so forth. If $\mathcal{G}$ is finite, then all messages can be computed in at most $\delta(\mathcal{G})$ rounds, where $\delta(\mathcal{G})$ is the maximum possible depth, called the *diameter*.

For any internal variable Y_j, we define the **marginal partition function** $Z_j(y_j)$ as

$$Z_j(y_j) = \overrightarrow{\mu}_j(y_j)\overleftarrow{\mu}_j(y_j), \quad y_j \in \mathcal{Y}_j.$$

Thus $Z_j(y_j)$ is simply the componentwise (dot) product of the messages $\overrightarrow{\mu}_j(y_j)$ and $\overleftarrow{\mu}_j(y_j)$. This is sometimes called the **past-future decomposition rule** [12].

Graphically, $Z_j(y_j)$ is the partition function of the graph obtained from $\mathcal{G}$ by converting Y_j from an internal to an external variable as shown in Figure 12; *i.e.,* by replacing the edge associated with Y_j by a "tap" consisting of the concatenation of an edge labeled by $\overrightarrow{Y}_j$, an equality indicator function, and another edge labeled by $\overleftarrow{Y}_j$, with a further half-edge labeled by Y_j attached to the equality indicator function.

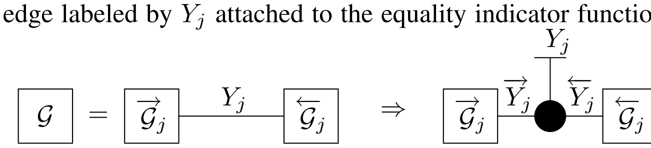

Fig. 12. Converting Y_j from internal to external by inserting a "tap."

Conversely, $Z(\mathcal{G})$ is the partition function of the graph obtained by converting Y_j back to an internal variable; *i.e.,* by summing $Z_j(y_j)$ over Y_j:

$$Z(\mathcal{G}) = \sum_{y_j \in \mathcal{Y}_j} Z_j(y_j) = \sum_{y_j \in \mathcal{Y}_j} \overrightarrow{\mu}_j(y_j)\overleftarrow{\mu}_j(y_j).$$

Thus, for any edge Y_j, $Z(\mathcal{G})$ is simply the dot product of the messages $\overrightarrow{\mu}_j$ and $\overleftarrow{\mu}_j$.

V. HOLOGRAPHIC TRANSFORMATIONS

In this section, we recapitulate and generalize the concept of "holographic transformations" of normal factor graphs, which was introduced by Al-Bashabsheh and Mao [2], and their "generalized Holant theorem," which relates the partition function of a normal factor graph to that of its holographic transform. This theorem generalizes the Holant theorem of Valiant [21] (see also [3], [4], [5], [6], [22]), which has been used to show that some seemingly intractable counting problems on graphs are in fact tractable.

Using this concept, Al-Bashabsheh and Mao [2] were able to prove a very general and powerful Fourier transform duality theorem for normal factor graphs, of which the original normal graph duality theorem of [12] is an immediate corollary. We give a variation of this proof which is perhaps even simpler (compare also the proof in [13]).

In the last section of this paper, we will sketch further applications of this general approach.

A. General approach

The general approach can be explained very simply, as follows. Let $\mathcal{A}$ and $\mathcal{B}$ be two finite alphabets, which will often be of the same size; *i.e.,* $|\mathcal{A}| = |\mathcal{B}|$. Let $U(a,b), S(b,b')$, and $V(b',a')$ be complex-valued factors involving variables A, B, B', and A' defined on $\mathcal{A}$, $\mathcal{B}$, $\mathcal{B}$, and $\mathcal{A}$, respectively; alternatively, we may regard U, S, and V as matrices. Finally, suppose that the concatenation USV, shown in Figure 13, is the identity factor $\delta_{aa'}$, which can be represented simply as an ordinary edge as in Figure 3.[5]

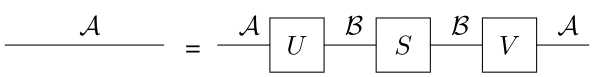

Fig. 13. A concatenation of factors that is equivalent to the identity.

We then have the following obvious lemma:

Lemma (generalized holographic transformations). In any NFG, any ordinary edge may be replaced by a concatenation of factors USV equivalent to the identity, as in Figure 13, without changing the partition function. □

The "holographic transformations" of [2] involve similar replacements, except without the middle factor S (alternatively, with $S(b,b') = \delta_{bb'}$). Al-Bashabsheh and Mao [2] call $\mathcal{B}$ the *coupling alphabet*, and say that U and V are *dual* with respect to $\mathcal{B}$. When $|\mathcal{A}| = |\mathcal{B}|$, they say that U and V are *transformers*; in this case, as matrices, U and V are inverses.

If a normal factor graph has external variables X_i, then they may be transformed as well, by the insertion of a factor or matrix $W_i(x_i, w_i)$ defined on $\mathcal{X}_i \times \mathcal{W}_i$, where $\mathcal{W}_i$ is the alphabet of a transformed external variable W_i. Thus the partition function is transformed into a function of the new external variables W_i. This is the essence of the "generalized Holant theorem" of [2]. (The original Holant theorem of Valiant [21] applies when there are no external variables.)

[5]Here and subsequently we may label an internal edge simply by its alphabet, without introducing dummy internal variables.

B. General normal factor graph duality theorem

This general approach yields a very simple proof of the "general normal factor graph duality theorem" of [2], [13].

Suppose that we have a normal factor graph in which each variable alphabet $\mathcal{A}$ is a finite-dimensional vector space over a finite field $\mathbb{F}$ of characteristic p (*i.e.,* p is the least positive integer such that $p\alpha = 0$ for all $\alpha \in \mathbb{F}$). The dual space $\hat{\mathcal{A}}$ is then a vector space over $\mathbb{F}$ of the same dimension as $\mathcal{A}$, and there is a well-defined $\mathbb{Z}_p$-valued *inner product* $\langle \hat{a}, a \rangle$ with the usual properties; *e.g.,* $\langle \hat{a}, 0 \rangle = \langle 0, a \rangle = 0$, $\langle \hat{a}, a + a' \rangle = \langle \hat{a}, a \rangle + \langle \hat{a}, a' \rangle$, and so forth (see, *e.g.,* [11]).

Given a complex-valued function $f : \mathcal{A} \to \mathbb{C}$ defined on $\mathcal{A}$, its *Fourier transform* is then defined as the complex-valued function $F : \hat{\mathcal{A}} \to \mathbb{C}$ on $\hat{\mathcal{A}}$ that maps $\hat{a}$ to

$$F(\hat{a}) = \sum_{a \in \mathcal{A}} f(a) \omega^{\langle \hat{a}, a \rangle}, \quad \hat{a} \in \hat{\mathcal{A}},$$

where $\omega = e^{2\pi i/p}$ is a primitive complex pth root of unity.

In an NFG, a Fourier transform may be represented as in Figure 14, where the Fourier transform factor is

$$\mathcal{F}_{\mathcal{A}} = \{\omega^{\langle \hat{a}, a \rangle} : \hat{a} \in \hat{\mathcal{A}}, a \in \mathcal{A}\}.$$

The transform $F(\hat{a})$ is obtained by summing over $\mathcal{A}$, which in this case amounts to a matrix-vector multiplication.

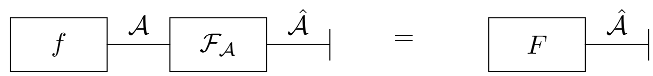

Fig. 14. Normal factor graph of a Fourier transform.

Note that as a factor in an NFG, we do not have to distinguish between $\mathcal{F}_{\mathcal{A}}$ and its transpose; $\mathcal{F}_{\mathcal{A}}$ is simply a function of the two variables corresponding to the two incident edges, and as a matrix can act on either variable. Thus $\mathcal{F}_{\mathcal{A}}$ can act also as a Fourier transform $\mathcal{F}_{\hat{\mathcal{A}}}$ on a function of $\hat{\mathcal{A}}$.

More generally, given a complex-valued multivariate function $f(\mathbf{a})$ defined on a set of variables $\mathbf{A} = \{A_i\}$ whose alphabets $\mathcal{A}_i$ are vector spaces over $\mathbb{F}$, its Fourier transform is defined as the complex-valued function

$$F(\hat{\mathbf{a}}) = \sum_{\mathbf{a}} f(\mathbf{a}) \prod_i \omega^{\langle \hat{a}_i, a_i \rangle}.$$

In other words, in a normal factor graph, each variable A_i may be transformed separately, as illustrated in Figure 15. In [2], this property is called *separability*.

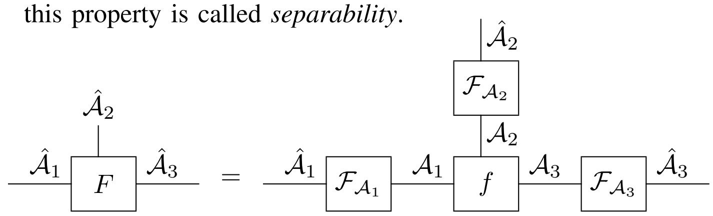

Fig. 15. Fourier transform of multivariate function $f(a_1, a_2, a_3)$.

Now let us define $U = V = \mathcal{F}_{\mathcal{A}}$ and $S = \Phi_{\sim}/|\mathcal{A}|$, where the *sign inverter indicator function* over $\hat{\mathcal{A}}$ is defined as

$$\Phi_{\sim}(\hat{a}, \hat{a}') = \begin{cases} 1, & \text{if } \hat{a} = -\hat{a}'; \\ 0, & \text{otherwise.} \end{cases}$$

Then the concatenation USV is the identity, since

$$\sum_{\hat{a}\in\hat{\mathcal{A}},\hat{a}'\in\mathcal{A}} \omega^{\langle\hat{a},a\rangle}\Phi_{\sim}(\hat{a},\hat{a}')\omega^{\langle\hat{a}',a'\rangle} = \sum_{\hat{a}\in\hat{\mathcal{A}}} \omega^{\langle\hat{a},a-a'\rangle} = |\mathcal{A}|\delta_{aa'},$$

by a basic orthogonality relation for Fourier transforms over finite groups (see, *e.g.,* [11]). This result is illustrated in Figure 16, where we omit the scale factor of $|\mathcal{A}|$.

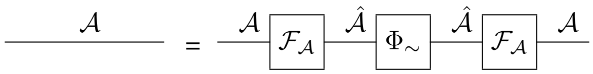

Fig. 16. A concatenation of factors that is equivalent to an edge, up to scale.

Now we can prove our desired result:

Normal factor graph duality theorem [2], [13]. Given an NFG with partition function $Z(\mathbf{x})$, comprising external variables X_i associated with half-edges, internal variables Y_j associated with ordinary edges (all alphabets being vector spaces over a finite field $\mathbb{F}$), and complex-valued factors f_k associated with vertices, the **dual normal factor graph** is defined by replacing each alphabet $\mathcal{X}_i$ or $\mathcal{Y}_j$ by its dual alphabet $\hat{\mathcal{X}}_i$ or $\hat{\mathcal{Y}}_j$, each factor f_k by its Fourier transform $\hat{f}_k$, and finally by placing a sign inverter indicator function $\Phi_{\sim}$ in the middle of every ordinary edge. Then the partition function of the dual NFG is the Fourier transform $\hat{Z}(\hat{\mathbf{x}})$ of $Z(\mathbf{x})$, up to scale.[6] □

Proof: Let us first convert the given NFG with partition function $Z(\mathbf{x})$ to an NFG with partition function $\hat{Z}(\hat{\mathbf{x}})$, up to scale, by appending a Fourier transform $\mathcal{F}_{\mathcal{X}_i}$ from $\mathcal{X}_i$ to $\hat{\mathcal{X}}_i$ to every half-edge associated with every external variable X_i, as in Figure 15. Then let us replace every ordinary edge associated with every internal variable Y_j by a concatenation $\mathcal{F}_{\mathcal{A}}\Phi_{\sim}\mathcal{F}_{\mathcal{A}}$ like that shown in Figure 16; this preserves the partition function $\hat{Z}(\hat{\mathbf{x}})$, up to scale. Now each vertex associated with each factor f_k is surrounded by Fourier transforms of all of the variables involved in f_k, so it and its surrounding transforms may be replaced by a single vertex representing the Fourier transform factor $\hat{f}_k$ without changing the partition function, up to scale. □

Notice that this remarkably general theorem applies to any normal factor graph, whether or not it has cycles.

Using the fact that the indicator functions of a linear code $\mathcal{C}$ over $\mathbb{F}$ and of its orthogonal code $\mathcal{C}^{\perp}$ are a Fourier transform pair, up to scale, one obtains as an immediately corollary a duality theorem for normal factor graph representations of linear codes [2], [13], which is equivalent to the original normal graph duality theorem of [12].

VI. Further Developments

We now sketch briefly how the "tree-based reparameterization" approach of Wainwright *et al.* [25], the "loop calculus" results of Chertkov and Chernyak [7], [8], and the Lagrange duality results of Vontobel and Loeliger [23], [24] fit within this generalized framework. The full developments will appear in a subsequent version of this paper.

[6] As shown in [2], the scale factor is $|\mathcal{Y}|$.

A. Tree-based reparameterization

Wainwright, Jaakkola, and Willsky [25] have shown how the sum-product algorithm applied to general graphs with cycles can be understood as a tree-based reparameterization algorithm, where each round of the message-passing algorithm reparameterizes marginal distributions over simple subtrees consisting of a pair of vertices connected by an edge. More generally, they consider iterative algorithms that reparameterize distributions over arbitrary cycle-free subtrees of the graph, particularly spanning trees.

Let $\mathbf{X}$ be a set of m variables X_i taking values x_i in finite alphabets $\mathcal{X}_i$, and let E be a set of pairs (X_i, X_j) indicating which pairs of variables are connected. Suppose that the corresponding graph with vertices X_i and edges $(X_i, X_j) \in E$ is a tree (*i.e.,* cycle-free). Finally, suppose that a probability distribution $p(\mathbf{x})$ over these variables can be expressed as

$$p(\mathbf{x}) \propto \prod_{1\le i\le m} \psi_i(x_i) \prod_{(X_i,X_j)\in E} \psi_{ij}(x_i, x_j),$$

where the functions $\psi_i(x_i)$ and $\psi_{ij}(x_i, x_j)$ depend only on the singleton variables X_i and pairs (X_i, X_j), respectively. (By the Hammersley-Clifford theorem, this can always be done when $p(\mathbf{x})$ is a positive Markov random field over the graph.)

We can view such a distribution $p(\mathbf{x})$ as a partition function in which all variables are external (a "global function"). Normalizing this partition function, we obtain an equivalent partition function with the same external variables, but with an equality indicator function corresponding to each external variable replacing it in the corresponding normal factor graph. A typical fragment of such an NFG is shown in Figure 17.

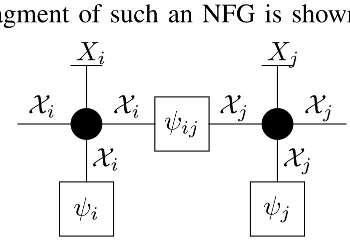

Fig. 17. Fragment of NFG representing a probability distribution on a tree.

Now we can execute the sum-product algorithm on such a cycle-free NFG, obtaining on each edge two messages, say $\overrightarrow{\mu}_i(x_i)$ and $\overleftarrow{\mu}_i(x_i)$ on an edge with alphabet $\mathcal{X}_i$. The corresponding marginal probability distribution $p_i(x_i)$ is proportional to the componentwise product of these messages:

$$p_i(x_i) \propto \overrightarrow{\mu}_i(x_i)\overleftarrow{\mu}_i(x_i), x_i \in \mathcal{X}_i.$$

Such a marginal distribution can be exhibited explicitly as a message in a "reparameterized" NFG by replacing a factor such as $\psi_{ij}(x_i, x_j)$ by the concatenation of three factors:

$$\begin{aligned} U(x_i, x_i') &= \overleftarrow{\mu}_i(x_i)\delta(x_i, x_i'); \\ S(x_i', x_j') &= \frac{\psi_{ij}(x_i', x_j')}{\overleftarrow{\mu}_i(x_i')\overrightarrow{\mu}_j(x_j')} \\ V(x_j, x_j') &= \overrightarrow{\mu}_j(x_j)\delta(x_j, x_j'), \end{aligned}$$

which evidently preserves the partition function.

Such a reparameterization can be performed also in a graph with cycles, or over a subtree of a given graph. Nice results are obtained when the messages are those that occur at a fixed point of the sum-product algorithm, but the messages do not have to be chosen in this way.

In future work, we plan to use this approach to restate and generalize many of the results of [25] and related papers.

B. Loop calculus

Chertkov and Chernyak [7], [8], [9] have developed a "loop calculus" for statistical systems defined on finite graphs that allows the partition function of a system to be expressed as a finite sum over "generalized loops," in which the lowest-order term corresponds to the Bethe-Peierls (sum-product algorithm) approximation.

We briefly sketch our approach to their results. Suppose that all alphabets are binary. Then replace every edge Y_j in the system by the concatenation $U_j S_j V_j$, where in matrix notation

$$\begin{aligned} U_j &= \begin{bmatrix} +\overleftarrow{\mu}_j(0) & -\overrightarrow{\mu}_j(1) \\ +\overleftarrow{\mu}_j(1) & +\overrightarrow{\mu}_j(0) \end{bmatrix}; \\ S_j &= \frac{1}{\Delta_j}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}; \\ V_j &= \begin{bmatrix} +\overrightarrow{\mu}_j(0) & +\overrightarrow{\mu}_j(1) \\ -\overleftarrow{\mu}_j(1) & +\overleftarrow{\mu}_j(0) \end{bmatrix}, \end{aligned}$$

where $\overleftarrow{\mu}_j(y_j)$ and $\overrightarrow{\mu}_j(y_j)$ are functions that may (but need not) be chosen as fixed-point messages of the sum-product algorithm, and $\Delta_j = \overrightarrow{\mu}_j(0)\overleftarrow{\mu}_j(0) + \overrightarrow{\mu}_j(1)\overleftarrow{\mu}_j(1)$ is the determinant of U_j and V_j. Evidently the concatenation $U_j S_j V_j$ is the identity, so this replacement preserves the partition function.

Now express every S_j as the sum of two matrices:

$$S_j = \frac{1}{\Delta_j}\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + \frac{1}{\Delta_j}\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix};$$

if there are n edges Y_j, then the partition function of the original NFG can correspondingly be expressed as the sum of the partition functions of the 2^n component NFGs.

If the functions $\overleftarrow{\mu}_j(y_j)$ and $\overrightarrow{\mu}_j(y_j)$ are fixed-point messages of the sum-product algorithm, then it turns out that the partition function of the "zero-order" component graph is the Bethe-Peierls partition function (at that fixed-point of the sum-product algorithm); that the partition function of any component graph with a "loose end" (a vertex of effective degree 1) is zero; and that the partition functions of the remaining component graphs (corresponding to "generalized loops," in which all vertices have effective degree 2 or more) are "small" multiples of the Bethe-Peierls partition function. Again, the full development will be given in a subsequent version of this paper.

C. Lagrange duality

Structurally similar operations can be used to obtain the Lagrange duality results for normal graphs of Vontobel and Loeliger [23], [24], which are based on the Legendre transform of convex optimization theory.

One interesting aspect of this development is that instead of sums of products, we consider minima over sums (*i.e.,* the sum-product semiring over the reals $\mathbb{R}$ is replaced by the min-sum semiring over the extended real line $\bar{\mathbb{R}} = \mathbb{R} \cup +\infty$). Thus a partition function has the following form:

$$Z(\mathbf{x}) = \min_{\mathbf{y} \in \mathcal{Y}} \sum_{k \in \mathcal{K}} f_k(\mathbf{x}_k, \mathbf{y}_k), \quad \mathbf{x} \in \mathcal{X},$$

where the "factors" $f_k(\mathbf{x}_k, \mathbf{y}_k)$ are $\bar{\mathbb{R}}$-valued.

The dual functions under the Legendre transform are functions in the max-sum semiring. Dualization involves the insertion of sign inverters into edges, as with Fourier dualization. Again, details will be provided in future versions of this paper.

Acknowledgment

We wish to acknowledge our close collaboration with Yongyi Mao, which led to the normal factor graph paradigm.

References

[1] S. M. Aji and R. J. McEliece, "The generalized distributive law," *IEEE Trans. Inf. Theory*, vol. 46, pp. 325-343, Mar. 2000.

[2] A. Al-Bashabsheh and Y. Mao, "Normal factor graphs and holographic transformations," *IEEE Trans. Inf. Theory*, pp. 752–763, Feb. 2011.

[3] J.-Y. Cai and V. Choudhary, "Valiant's Holant theorem and matchgate tensors," *Theoretical Computer Science*, vol. 384, pp. 22–32, 2007.

[4] J.-Y. Cai and P. Lu, "Holographic algorithms: From art to science," in *Proc. 39th Annual ACM Symp. on Theory of Computing* (San Diego, CA), pp. 401–410, June 2007.

[5] J.-Y. Cai, P. Lu, and M. Xia, "Holographic algorithms by Fibonacci gates and holographic reductions for hardness," in *Proc. IEEE 49th Annual IEEE Symp. on Foundations of Computer Science* (Philadelphia, PA), pp. 644–653, October 2008.

[6] J.-Y. Cai, P. Lu, and M. Xia, "Holographic algorithms with matchgates capture precisely tractable planar #CSP," Aug. 2010. ArXiv: 1008.0683.

[7] M. Chertkov and V. Y. Chernyak, "Loop calculus in statistical physics and information science," *Phys. Rev. E*, vol. 73, no. 065102(R), 2006.

[8] M. Chertkov and V. Y. Chernyak, "Loop series for discrete statistical models on graphs," *J. Stat. Mech.*, P06009, 2006.

[9] V. Y. Chernyak and M. Chertkov, "Planar graphical models which are easy," *J. Stat. Mech.*, P11007, Nov. 2010.

[10] P. Cvitanović, *Group Theory: Birdtracks, Lie's, and Exceptional Groups*. Princeton U. Press, 2008. [On-line] birdtracks.eu.

[11] G. D. Forney, Jr., "Transforms and groups," in *Codes, Curves and Signals: Common Threads in Communications* (A. Vardy, ed.), pp. 79–97. Boston: Kluwer, 1998.

[12] G. D. Forney, Jr., "Codes on graphs: Normal realizations," *IEEE Trans. Inf. Theory*, vol. 47, pp. 520–548, Feb. 2001.

[13] G. D. Forney, Jr., "Codes on graphs: Duality and MacWilliams identities," to appear, *IEEE Trans. Inf. Theory*, vol. 57, Mar. 2011.

[14] F. R. Kschischang, B. J. Frey and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, pp. 498–519, Feb. 2001.

[15] H.-A. Loeliger, “An introduction to factor graphs,” *IEEE Sig. Proc. Mag.*, vol. 21, pp. 28–41, Jan. 2004.

[16] H.-A. Loeliger, J. Dauwels, J. Hu, S. Korl, L. Ping and F. R. Kschischang, “The factor graph approach to model-based signal processing,” *Proc. IEEE*, vol. 95, pp. 1295–1322, June 2007.

[17] S. Morse and E. Peterson, “Trace diagrams, matrix minors, and determinant identities.” ArXiv: 0903.1373.

[18] R. Penrose, *The Road to Reality: A Complete Guide to the Laws of the Universe*. New York: Knopf, 2005.

[19] E. Peterson, “Unshackling linear algebra from linear notation.” ArXiv: 0910.1362.

[20] G. E. Stedman, *Diagrammatic Techniques in Group Theory.* Cambridge U. Press, 1990.

[21] L. G. Valiant, “Holographic algorithms,” in *Proc. 45th Annual IEEE Symp. on Foundations of Computer Science* (Rome, Italy), pp. 306–315, October 2004.

[22] L. G. Valiant, “Some observations on holographic algorithms,” in *Proc. 9th Latin American Theoretical Informatics Symp.* (Oaxaca, Mexico), pp. 577–590, April 2010.

[23] P. O. Vontobel, *Kalman Filters, Factor Graphs, and Electrical Networks*, post-diploma project, ETH–Zurich, 2002. [On-line]: http://www.isiweb.ee.ethz.ch/papers.

[24] P. O. Vontobel and H.-A. Loeliger, “On factor graphs and electrical networks,” in *Mathematical Systems Theory in Biology, Communication, Computation and Finance* (J. Rosenthal and D. Gilliam, eds.), IMA Volumes in Math. and Appl., vol. 134, pp. 469–492. New York: Springer-Verlag, 2003.

[25] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky, ”Tree-based reparameterization framework for analysis of sum-product and related algorithms,” *IEEE Trans. Inf. Theory*, vol. 49, pp. 1120–1146, May 2003.

[26] N. Wiberg, H.-A. Loeliger, and R. Kötter, “Codes and iterative decoding on general graphs,” *European Transactions on Telecommunications*, vol. 6, pp. 513–525, Sept. 1995.