A Hypergraph-based Reduction for Higher-order Binary Markov Random Fields

Alexander Fix, Aritanan Gruber, Endre Boros, Ramin Zabih

Abstract—Higher-order Markov Random Fields, which can capture important properties of natural images, have become increasingly important in computer vision. While graph cuts work well for first-order MRF's, until recently they have rarely been effective for higher-order MRF's. Ishikawa's graph cut technique [1], [2] shows great promise for many higher-order MRF's. His method transforms an arbitrary higher-order MRF with binary labels into a first-order one with the same minima. If all the terms are submodular the exact solution can be easily found; otherwise, pseudoboolean optimization techniques can produce an optimal labeling for a subset of the variables. We present a new transformation with better performance than [1], [2], both theoretically and experimentally. While [1], [2] transforms each higher-order term independently, we use the underlying hypergraph structure of the MRF to transform a group of terms at once. For n binary variables, each of which appears in terms with k other variables, at worst we produce n non-submodular terms, while [1], [2] produces $O(nk)$. We identify a local completeness property under which our method perform even better, and show that under certain assumptions several important vision problems (including common variants of fusion moves) have this property. We show experimentally that our method produces smaller weight of non-submodular edges, and that this metric is directly related to the effectiveness of QPBO [3]. Running on the same field of experts dataset used in [1], [2] we optimally label significantly more variables (96% versus 80%) and converge more rapidly to a lower energy. Preliminary experiments suggest that some other higher-order MRF's used in stereo [4] and segmentation [5] are also locally complete and would thus benefit from our work.

1 Introduction

Higher-order MRF's have become increasingly important in computer vision, as they can incorporate sophisticated priors such as fields of experts (FoE) [7]. Performing inference in higher-order MRF's, however, requires minimizing significantly more complex energy functions than for standard first-order models. Graph cuts are a popular method for solving first-order MRF's, such as the benchmarks described in [8] and [9], but are much more difficult to apply to higher-order MRF's. As a result, until recently this powerful optimization method has only been used for a few specialized higher-order MRF's, such as [4], [10].

Ishikawa [1], [2] proposed a general-purpose graph cut technique for arbitrary higher-order MRF's that shows great promise. This method works by transforming the higher-order input MRF into an equivalent quadratic (pairwise) MRF by adding additional auxiliary variables and edges. The general class of such methods are known as *higher-order reductions* — this particular reduction is commonly referred to as Higher-Order Clique Reduction (HOCR). As an application he considered fields of experts for denoising, an important higher-order MRF that is typically solved by belief propagation (BP) [11]. Prior to [1], [2] there was no graph cut method that could successfully handle such an energy function. Ishikawa's graph cut method produced better results than the BP method of [11], and ran an order of magnitude faster [1], [2].

A very different approach to finding higher-order reductions is Generalized Roof Duality (GRD) [12], [13], which proposed a class of submodular relaxations for an arbitrary higher-order MRF with degree at most 4. GRD finds the best such relaxation by solving a linear program, and optimizes the relaxed function exactly using graph cuts. The relaxations found by GRD provide very good approximate solutions to the original higher-order problem. Beyond the restriction on the MRF's degree, which appears difficult to overcome, GRD is also computationally much more intensive than HOCR.

In this paper we propose an alternative construction to HOCR and GRD, with improved theoretical and experimental performance. Instead of considering terms in the energy function one at a time, we make use of the fact that the clique structure of an MRF is a hypergraph, in order to reduce many terms at once. We review existing methods for solving higher-order MRF's with graph cuts in section 2, focusing on methods that directly compete with our work [1], [2], [12], [13], [14], [15], [16], [17]; the comparison is summarized in figure 1. We present our new algorithm in section 3, and analyze its worst case performance in section 4. In section 5 we show that for problems with property called *local completeness* our method performs even better. Under certain assumptions we prove that some important vision problems are locally complete, including the fields of experts MRF considered by Ishikawa. Experimental results are given in section 7, along with experimental evidence that other vision problems [4], [5] are also locally complete.

Alexander Fix and Ramin Zabih are with the Cornell Computer Science Department. Aritanan Gruber and Endre Boros are with the Rutgers Center for Operations Research. E-mail: afix@cs.cornell.edu, Aritanan.Gruber@gmail.com, Endre.Boros@rutgers.edu, rdz@cs.cornell.edu. *A preliminary version of this work appeared in [6].*

	New variables	Non-submodular terms	Submodular terms	Non-submodular weight
Substitution [16] §2.3.1	$O(nk)$	$O(nk)$	$O(nk)$	$O(nkM)$
Negative [14] §2.3.2	t	–	td	–
HOCR [1] §2.3.3	$O(td)$	$O(nk)$	$O(td^2)$	$O(d^2W)$
GRD [12] ($d \leq 4$) §2.3.4	$n + O(t)$	–	$O(td)$	–
Ours (worst case)	$n + O(td)$	n	$O(td^2)$	$O(dW)$
Ours (local completeness)	$n + O(t)$	n	$O(td)$	$O(dW)$

Fig. 1. Resources required to reduce t terms of degrees up to d, for an energy function with n variables each of which occurs with up to k other variables. W is the total weight of all positive terms in the higher-order function. Unlike the other algorithms listed, GRD is only defined for terms of limited degree. There is no clear notion of non-submodular edges in the relaxation produced by GRD, so we mark these entries "–". Non-submodular weight is the total weight of non-submodular edges in the reduced function.

2 Related Work

2.1 Graph cuts and QPBO

Graph cut methods solve energy minimization problems by constructing a graph and computing the min cut. The most popular graph cut methods, such as the expansion move algorithm of [18], repeatedly solve an optimization problem over binary variables. Such problems have been extensively studied in the operations research community, where they are referred to as pseudoboolean optimization [19] (see [20] for a more detailed survey).

Minimizing an arbitrary binary energy function is NP-hard (see e.g. [15]). Certain binary optimization problems can be solved exactly by min cut; these include computing the optimal expansion move [18] and its generalization to fusion moves [21], as well as a number of other interesting problems such as [10]. The recent comparison of [9] has shown expansion moves to be very effective at obtaining fast, approximate solutions for a wide range of vision problems.

The most important class of optimization problems that can be solved exactly with graph cuts are quadratic functions where every term is submodular [15], [20], [22]. Such a function can be written as a polynomial in the binary variables $(x_1, \ldots, x_n) \in \mathbb{B}^n$ of the form

$$\sum_i \alpha_i x_i + \sum_{i,j} \alpha_{i,j} x_i x_j. \tag{1}$$

This function is submodular if and only if $\alpha_{i,j} \leq 0$ for every i, j.

The most widely used technique for minimizing binary energy functions with non-submodular terms relies on the roof duality technique of [23] and the associated graph construction [20]. This method, commonly referred to as QPBO [3], uses min cut to compute the global minimum for any submodular energy function and for certain other functions as well (see [3], [20] for a discussion). Even when QPBO does not compute the global minimum, it provides a partial optimality guarantee called persistency [23]; QPBO computes a partial assignment which, when applied to an arbitrary complete assignment, will never cause the energy to increase. Such a partial assignment gives the variables it labels their values in the global minimum. Efficient techniques for computing persistencies, along with generalizations and implementations were proposed in [3], [20].

2.2 Higher-order MRF's

Higher-order MRF's, have recently become popular, especially due to models such as fields of experts [7]. Message passing approaches, such as belief propogation or dual decomposition [11], [24], can be used for higher-order MRF's, but their efficiency is a concern. For first-order MRF's, the performance of graph cut methods is competitive with message passing, and sometimes exceeds it [8]. This has lead to significant interest in applying graph cuts to higher-order MRF's.

QPBO is among the most widely used graph cuts algorithms, but it is restricted to handling energy functions which are quadratic functions of binary variables. Move making algorithms such as expansion moves [18] or fusion moves [21], which are very successful on first-order MRF's, generate a series of binary optimization problems. These moves can also be applied to higher-order MRF's, and the resulting binary optimization problem involves a multilinear polynomial instead of a quadratic one, i.e., a function of the form

$$\sum_{H \subseteq V} \alpha_H \prod_{i \in H} x_i \tag{2}$$

The the main technical challenge with higher-order MRF's is to reduce this multilinear polynomial into quadratic form; if the resulting quadratic is submodular it can be solved exactly, while otherwise QPBO can be used to identify persistencies.

It is preferable to identify moves for higher-order MRF's where the binary optimization problem can be reduced to a submodular quadratic one and thus solved exactly. This approach has proven successful for some cases, notably the $\mathcal{P}^n$ model of [10]. However, this cannot be done for general higher-order MRF's — [25] showed that there are submodular functions with cliques of size 4 which cannot be reduced to a submodular quadratic form, and non-submodular functions can never be reduced to a submodular quadratic function. Therefore,

for arbitrary higher-order MRF's the natural approach is reduction to a quadratic followed by QPBO.

For a more complete discussion of inference methods in MRF's (including a discussion of other higher-order methods) see the survey [26].

2.3 Reduction techniques

There are a number of methods for reducing an arbitrary multilinear polynomial over binary variables into a quadratic one. The performance of the different methods is summarized in figure 1.

For all methods, we are interested in the size of the obtained quadratic function, including the number of additional vertices and edges required, as these directly affect the size of the min cut problem which will be solved by QPBO. We make a particular note of the number of nonsubmodular edges as well as the weight of these edges, as these can negatively impact the solution returned by QPBO [8], as confirmed in our experiments[1].

2.3.1 Reduction by substitution

The original reduction was introduced by Rosenberg [16]. The reduction eliminates all occurrences of some product xy by introducing a new variable z, replacing xy by z everywhere it occurs, and then adding the following penalty terms to the energy function: $Mxy - 2Mxz - 2Myz + 3Mz$, where M is a suitably large constant. This forces z to take the value of xy in any optimal solution.

If each variable is in terms with at most k other variables, this reduction can be done with $O(nk)$ pairs, which results in $O(nk)$ new variables, non-submodular terms and submodular quadratic terms.

Note that the non-submodular terms have large coefficients. Experimentally it has been reported that QPBO performs very poorly on such energy functions (see, for example, [2, §8.3.4], which states that QPBO finds almost no persistencies).

2.3.2 Reducing negative-coefficient terms

Kolmogorov and Zabih [15] for $d = 3$ and Freedman and Drineas [14] for $d \geq 3$ suggested the following transformation for negative higher degree terms:

$$-x_1 \cdots x_d = \min_{y \in \mathbb{B}} \; y\left((d-1) - \sum_{j=1}^{d} x_j\right) \qquad (3)$$

If we have t negative-coefficient terms of degree d, this gives t new variables and td submodular quadratic terms, but no new non-submodular terms.

Let us note that the above equality remains valid even if we replace some of the x_j variables with their complements $\overline{x}_j = (1 - x_j)$. Rother [17] suggested recently a transformation equivalent with this observation (see type-II transformations in [17]) together with a new transformation (see type-I in [17]).

2.3.3 Reducing positive-coefficient terms

The HOCR transformation [1], [2] was the first practical for general higher-order functions. For a term of degree d, let $n_d = \lfloor \frac{d-1}{2} \rfloor$ and set $c_{i,d} = 1$ if $d = i$ is odd, and $c_{i,d} = 2$ otherwise. Each positive term is reduced by

$$x_1 \cdots x_d = \min_{u_1,\cdots,u_d} \sum_{i=1}^{n_d} u_i \left(c_{i,d}\left(-\sum_{j=1}^{d} x_j + 2i\right) - 1 \right) + \sum_{i<j} x_i x_j$$

For each term of degree d, we get $O(d)$ new variables. Each new variable is connected to each original variable by a submodular edge, for a total of $O(d^2)$ submodular edges. We also get non-submodular edges between all pairs of original variables x_i, x_j whenever x_i and x_j are in the same clique. If each variable occurs in terms with at most k other variables, then the number of non-submodular edges is $O(nk)$ (note that if the pair x_i, x_j occurs in multiple cliques, we only count the pair once, as they can be combined to a single edge in the flow network). Finally, if the positive term has positive weight $\alpha > 0$ then this term creates $\frac{d(d-1)}{2}$ non-submodular edges of weight α. So, if the total weight of positive terms is W, then the quadratic function has non-submodular edges of total weight $O(d^2 W)$.

Note that this reduction uses a large number of non-submodular edges: the d original variables are fully connected by positive weight edges. This is problematic, as it has been observed [8] that non-submodular edges can result in poor performance for graph cut optimizers like QPBO.

2.3.4 Generalized Roof Duality

Unlike the above methods, which are all rewrite rules for the individual terms of the multilinear polynomial, Generalized Roof Duality (GRD) [13] finds a reduction to quadratic form which is globally the best among a large class of candidates, called *submodular relaxations*.

GRD uses a characterization of all submodular functions expressible in quadratic form due to Zivny et. al. [25]. This reduction uses one additional variable for each term, as well as $O(d)$ additional edges for a degree d term.

The reduction also ensures the existence of *persistencies*, similar to the well-known persistencies of Roof Duality (better known as QPBO) whereby after solving the submodular relaxation, each variable is assigned a value in $\{0, 1, 1/2\}$, and every variable taking value 0 or 1 in the partial labeling actually takes that value in the (possibly unknown) global optimum.

To find the best submodular relaxation, GRD solves a linear program. Because GRD finds the tightest submodular relaxation, the returned labeling is typically of very high quality; however, solving the LP is computationally very expensive, making this algorithm impractical for large-sized problems. Instead of directly solving the LP,

1. Note that the total weight of nonsubmodular edges is not a perfect measure of the performance of QPBO. For example, many functions can have non-submodular edges, but after permuting some labels become submodular [27], and these functions can be exactly minimized by QPBO. Nevertheless, our experiments show this is a useful heuristic.

the authors also give heuristics to find nearly optimal submodular relaxations. These heuristic relaxations (denoted GRD-heur) also give very good labelings. However, we will show in the experimental section that even these heuristic methods are several times slower than HOCR and our technique.

Finally, it is worth noting that GRD can only be applied when all terms have degree 3 or 4, and it is doubtful that the method can be generalized. The reduction [25] used by GRD to convert the submodular relaxation to quadratic form has only been described for functions of arity 4. Furthermore, writing down an LP for the optimal submodular relaxation requires being able to compactly describe the set of submodular functions with terms of degree d, and this task is NP hard for $d \geq 4$. In contrast, our method and Ishikawa's have no restriction on the degree of terms involved.

3 Reducing Groups of Higher-Order Terms

The terms of a multilinear polynomial form a hypergraph $\mathcal{H}$. The vertices are the polynomial's variables, and there is a hyperedge $H = \{x_1, \ldots, x_d\}$ with weight α_H whenever the polynomial has a term $\alpha_H x_1 \cdots x_d$. In contrast to earlier methods which reduce term-by-term, our new method uses this hypergraph structure to reduce a group of terms all at once.

The two theorems below are both concerned with reducing respectively all the positive or all the negative terms containing a single variable, (or small set of variables); we will write this common subset of variables as C. The most important special case of our reduction is shown in figure 2, where we consider positive terms containing the single variable $C = \{x_1\}$.

Theorem 3.1: Let $\mathcal{H}$ be a set of terms such that for all $H \in \mathcal{H}$ the common set $C \subseteq H$. Furthermore, all the hyperedges H have positive weights $\alpha_H > 0$. Let $f(x) = \sum_{H \in \mathcal{H}} \alpha_H \prod_{j \in H} x_j$ be this polynomial. Then $f(x)$ is equal to

$$\min_{y \in \{0,1\}} \left(\sum_{H \in \mathcal{H}} \alpha_H \right) y \prod_{j \in C} x_j + \sum_{H \in \mathcal{H}} \alpha_H \overline{y} \prod_{j \in H \setminus C} x_j. \quad (4)$$

Proof: Given any assignment of the variables $x_1, \ldots, x_n$, either (1) all the variables in C are 1, or (2) some variable in C is 0.

Case 1: Subsituting 1 for the variables in C, $f(x)$ is equal to $\sum_{H \in \mathcal{H}} \alpha_H \prod_{j \in H \setminus C} x_j$ and (4) is $\min_y (\sum_{H \in \mathcal{H}} \alpha_H) y + \sum_{H \in \mathcal{H}} \alpha_H \overline{y} \prod_{j \in H \setminus C} x_j$. If we assign $y = 1$, then (4) becomes $\sum_{H \in \mathcal{H}} \alpha_H$, and if we assign $y = 0$, then it becomes $\sum_{H \in \mathcal{H}} \alpha_H \prod_{j \in H \setminus C} x_j$. This quantity is always less than or equal to $\sum_{H \in \mathcal{H}} \alpha_H$, so the minimum is achieved when $y = 0$, in which case, $f(x)$ equals (4).

Case 2: The product $\prod_{j \in C} x_j$ is 0. Since all the terms of $f(x)$ share the common subset C, $f(x) = 0$. Similarly, (4) is $\sum_{H \in \mathcal{H}} \alpha_H \overline{y} \prod_{j \in H \setminus C} x_j$. If we assign $y = 1$, then this sum is 0, whereas if we assign $y = 0$, then it is positive, since each α_H is positive. Thus, the minimum is achieved when $y = 1$, in which case (4) is 0 hence equal to $f(x)$. □

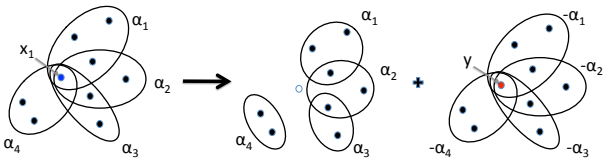

Fig. 2. Our main reduction. At left are all the original positive terms containing the common variable x_1 (so $\alpha_i > 0$). At right are all the new terms we obtain from equation (4). The positive terms on top are just the original terms minus x_1, and the negative terms on bottom are the original terms with y replacing x_1.

For every positive term containing the common subset C, equation (4) replaces it with a new term $\alpha_H \overline{y} \prod_{j \in H \setminus C} x_j$. To get a multilinear polynomial, we replace the negated variable $\overline{y}$ with $1 - y$, which splits each term into two: $\alpha_H \prod_{j \in H \setminus C} x_j$ and $-\alpha_H y \prod_{j \in H \setminus C} x_j$.

Corollary 3.2: When we apply equation (4) to a positive term, we obtain a positive term of smaller degree, and a negative term with y replacing the common subset C.

For reducing the negative-coefficient terms all sharing some common subset, we have a similar theorem.

Theorem 3.3: Consider $\mathcal{H}$ and C as above, where now the coefficients α_H are negative for all H. Let g be the corresponding polynomial. Then for any assignment of the variables, $g(x)$ is

$$\min_{y \in \{0,1\}} \sum_{H \in \mathcal{H}} -\alpha_H \left(1 - \prod_{j \in C} x_j - \prod_{j \in H \setminus C} x_j \right) y \quad (5)$$

Proof: The proof is similar to the proof for Theorem 3.1. The minimum is achieved when $y = \prod_{j \in C} x_j$. □

A crucial difference between this reduction and theorem 3.1 is that in the positive case, we could let the common subset C be a single variable. Doing this here removes the term $\alpha_H \prod_{j \in H} x_j$ and replaces it with $\alpha_H y \prod_{j \in H \setminus \{1\}} x_j$, another negative term of the same degree. Trying to apply this reduction repeatedly will thus never terminate. However, if C consists of two or more variables, then grouping all terms containing C and reducing results in smaller degree terms replacing every term that we start with.

3.1 Our method

Equations (4) and (5) can be used for different reduction strategies. Both depend upon the choice of common variables C. Besides choosing $|C|$, we can also decide the order to consider different choices of C; for example,

which single variable to use to apply equation (4), or which pair of variables to use to apply equation (5).

We will focus on the simplest case: we let the common part C be a single variable x_i, and reducing positive terms containing this variable via equation (4). Negative terms will be reduced using the method of section 2.3.2. Note that more complicated schemes are also possible, such as picking pairs of variables and reducing both positive and negative terms containing this pair via equations (4) and (5).

Our method reduces a multilinear polynomial with higher-order terms, to quadratic form in two steps:

Step 1. Eliminate all higher-order positive terms by repeated application of Theorem 3.1, with the common subset C set to a single variable x_1. Gather all terms containing x_1, and replace them with equation (4). If $\mathcal{H}$ consists of all positive terms containing x_1, then

$$\sum_{H\in\mathcal{H}} \alpha_H \prod_{j\in H} x_j = \min_{y\in\{0,1\}} \left(\sum_{H\in\mathcal{H}} \alpha_H \right) x_1 y \tag{6a}$$

$$+ \sum_{H\in\mathcal{H}} \alpha_H \prod_{j\in H\setminus\{1\}} x_j \tag{6b}$$

$$- \sum_{H\in\mathcal{H}} \alpha_H y \prod_{j\in H\setminus\{1\}} x_j \tag{6c}$$

The positive terms now form a hypergraph on one fewer variable, so repeat with $x_2, \ldots, x_n$ until all positive terms are reduced.

Step 2. All higher-order terms now have negative coefficients. Reduce them term-by-term using the methods in section 2.3.2.

Note that equation (6) is simply the special case of equation (4) for a single variable. This special case is illustrated in figure 2.

4 Worst Case Performance

The results of applying equation (6) consist of three parts: a positive quadratic term (6a); and for each term, a positive term on the same variables except x_1 (6b); and a negative term with y replacing x_1 (6c).

Note that in the course of the reduction, we may create a monomial on some variables $x_1, \ldots, x_d$ and that another monomial on the same variables already existed in the input. In this case, we get lucky, since we can just sum the new term's coefficient with the existing monomial, which doesn't increase the size of the representation. To analyze the worst-case performance, we will assume that this never happens. In section 5 we will revisit this possibility.

Under this assumption, each positive term of degree d that we start with will have a single variable removed every time we apply the reduction, so to be fully reduced it must go through $d-1$ applications of the rule, producing negative terms of degrees $2, \ldots, d$. Reducing these $d-1$ negative terms by section 2.3.2 results in $O(d)$ new variables and $O(d^2)$ submodular quadratic terms.

Overall, to reduce t positive terms of degree d on n variables, in the worst case our method requires $n+O(td)$ new variables, $O(td^2)$ submodular terms and at most n non-submodular terms. Even in the worst case our algorithm's asymptotic performance is similar to HOCR (see figure 1). However, our method produces at most n non-submodular terms, compared to $O(nk)$ for HOCR.

For the weight of non-submodular edges, each positive term contributes α_H to (6a) each time it is reduced, for a total of $(d-1)\alpha_H$ weight in non-submodular edges. If the total weight of positive terms is W, we get $O(dW)$ non-submodular weight in the reduced form, a factor d improvement over HOCR.

5 Local Completeness

We can improve on this worst-case analysis for some common vision problems such as [4], [7]. We have identified a property of certain energy functions that we call local completeness, where our algorithm (unlike HOCR or GRD) has improved asymptotic performance. The basic idea behind local completeness is that whenever a monomial $x_1 \cdots x_d$ occurs in the input, we are also likely to see all the monomials on all subsets of these d variables as well. In essence, local completeness argues that typical inputs to vision problems are "bad" in the sense of having lots of terms. If a certain problem is locally complete, then this is a lower-bound argument to show that the input is necessarily large, and hence methods (such as ours) which exploit the shared structure of the graph will be more successful.

To be precise, consider a multilinear polynomial on the binary variables $x_1, \ldots, x_n$, and denote by $\mathcal{H}$ the hypergraph of its monomials, as before. Note that $\mathcal{H}$ is not necessarily in minimal form (i.e., if $H \subseteq H'$ then both H and H' may both be hyperedges in $\mathcal{H}$). Let $\mathcal{H}'$ be the "completed" hypergraph, formed by all subsets of edges in $\mathcal{H}$ (that is, $\mathcal{H}' = \bigcup_{H\in\mathcal{H}} 2^H$).

Definition 5.1: We say that the polynomial is *locally complete with completeness* c (or has local completeness c) if there is a number $c \in (0, 1]$ such that $|\mathcal{H}| \geq c|\mathcal{H}'|$.

To explain the terminology, note that the larger hypergraph $\mathcal{H}'$ is obtained by *completing* our input $\mathcal{H}$, to include all the subsets of every term that we started with.

Every polynomial is locally complete for some completeness c, as we can always choose $c = \frac{|\mathcal{H}|}{|\mathcal{H}'|}$. However, we are interested in classes of problems which remain complete as the problem size grows, so we say that *a family of polynomials is locally complete* if there is a fixed c such that all the polynomials have local completeness c. For example, a family P of polynomials arising from a particular vision problem would be locally complete if we always had $1/2$ of all subsets of terms appearing in all instances of P.

5.1 Performance on locally complete problems

Recall the procedure for reducing positive terms, using equation 6. We would like the extra positive terms we

create, with variables in $H \setminus \{1\}$, to combine with existing terms. If it happens that $H \setminus \{1\}$ is already a term with coefficient $\beta_{H\setminus\{1\}}$, then we add α_H to this coefficient, and do not create a new term.

This motivates the definition of local completeness: the new positive terms in (6b) have variables which are subsets of our original terms, so if our energy function has local completeness c, the new positive terms will combine with existing terms fraction c of the time.

Theorem 5.2: If an energy function has local completeness c, our procedure for reducing positive terms will result in at most $\frac{1}{c}|\mathcal{H}|$ negative coefficient terms

Proof: By the definition of local completeness $|\mathcal{H}'| \leq \frac{1}{c}|\mathcal{H}|$. As a notational convenience, add in all the extra subsets contained in $\mathcal{H}'$ as monomials with coefficient 0, so there are now $|\mathcal{H}'|$ terms. Having done this, since $\mathcal{H}'$ is closed under subsets, the positive terms produced by (6b) will always combine with existing terms.

Applying equation 6 removes the term $\alpha_H \prod_{j\in H} x_j$, changes the coefficient on the term with variables $H \setminus \{1\}$, and adds a new negative term $\alpha_H y \prod_{j\in H\setminus\{1\}} x_j$. The total number of terms remains constant.

Therefore, when we have finished reducing all positive terms, we are left with only negative terms, and we have as many as we started with, namely $|\mathcal{H}'| \leq \frac{1}{c}|\mathcal{H}|$. □

If we started with t terms of up to degree d on n variables, the entire reduction results in at most $n + \frac{1}{c}t$ new variables, $\frac{1}{c}td$ submodular terms and n non-submodular terms. For a family of locally complete inputs, c is constant, giving the asymptotic results in figure 1.

Local completeness is stronger than strictly necessary — we only really need that when reducing terms containing x_1, all the terms $H \setminus \{1\}$ already exist. In this case, the analysis depends on the order in which we pick variables. Local completeness gives the stronger property that no matter what order we choose variables to reduce, we always get a large fraction of terms combining.

Finally, we would like to reiterate that local completeness does not state that such problems are easy to solve, but rather the opposite: such problems (which include many vision problems, as shown below) have intrinsically large representations as multilinear polynomials; but nevertheless, in such cases our reduction uses few additional variables and edges.

6 Locally Complete Energy Functions in Vision

We can show that under some reasonable assumptions an important class of vision problems will have locally complete energy functions. Specifically, we consider fusion moves [21] under an FoE prior [7] with random proposals, used as a benchmark for HOCR in [1], [2].

The original (non-binary) energy function can be written as a sum over cliques $\mathcal{C}$ in the image $\sum_C f_C(\mathbf{x}_C)$. A single fusion move has an input image I and a proposed image I', and for every pixel there is binary variable that encodes whether that pixel takes its intensity from I or I'. This results in a binary energy function on these variables to compute the optimal fusion move.

We can better analyze fusion moves by moving to a continuous framework. Embed the original intensities in $\mathbb{R}$, and extend the clique energies f_C to functions on $\mathbb{R}^d$. We need two assumptions: (1) f_C is $d-1$ times continuously differentiable and (2) each of the d different mixed partials $\frac{\partial^{d-1} f}{\partial x_1 \cdots \widehat{\partial x_i} \cdots \partial x_d}$ (where $\widehat{\partial x_i}$ means to omit the i-th partial) take their zeros in a set of measure 0.

Theorem 6.1: Under these two assumptions, the set of proposed-current image pairs (I, I') for which the fusion move binary energy function does not have local completeness 1 has measure 0 as a subset of $\mathbb{R}^n \times \mathbb{R}^n$.

We defer the proof of this theorem to the supplemental material, but now provide a proof sketch. We write the fusion move binary energy function in terms of n binary variables b_i. Writing this as a multilinear polynomial in b, each clique C can result in terms t_S for each subset S of C. We can show that the energy function is locally complete, if the coefficient on t_S is never (or rarely) zero.

For example, here is how to calculate the coefficient on the term $b_1 b_2$ in a clique of size 3. If I_1, I_2, I_3 are the labellings in the current image on C, and I'_1, I'_2, I'_3 are the proposed labellings, then the coefficient on $b_1 b_2$ is

$$f_C(I_0, I_1, I_2) - f_C(I'_0, I_1, I_2) - f_C(I_0, I'_1, I_2) + f_C(I'_0, I'_1, I_2)$$

Since the labels are in $\mathbb{R}$, the four 3-pixel images mentioned in this coefficient lie on a rectangle in $\mathbb{R}^3$. If we give each of these points v heights of $f_C(v)$, then this coefficient is 0 if and only if the four points are coplanar.

In general, we do not expect 4 arbitrary points to lie on a plane. However, if f_C has no curvature, then any 4 such points will be coplanar. In the full proof, we show that if there exists any open ball of image pairs with zero coefficient on $b_1 b_2$, then the energy function is flat ($\frac{\partial^2 f}{\partial x_1 \partial x_2} = 0$) in the same ball (contradicting our assumption that the partials are nonzero almost everywhere). We also extend this to larger degree terms, to prove the general case.

Corollary 6.2: The energy functions obtained from fusion moves with FoE priors and proposals chosen as random images are locally complete with probability 1.

Proof: The functions f_C for the FoE model given in [7] are infinitely differentiable, and their mixed partials have their zeros in a set of measure 0. Since the proposed images are chosen from a continuous distribution over $\mathbb{R}^n$, events of measure 0 occur with probability 0. □

7 Experimental Results

We have provided a freely available open source implementation of our algorithm at http://www.cs.cornell.edu/~afix/software.html. We experimentally compared our reduction with the two available, general-purpose higher-order reductions: HOCR [1], [2] and GRD [12], [13]. These three methods are all direct competitors in the types of energy functions they can handle (with the limitation that GRD is restricted to $d \leq 4$). All

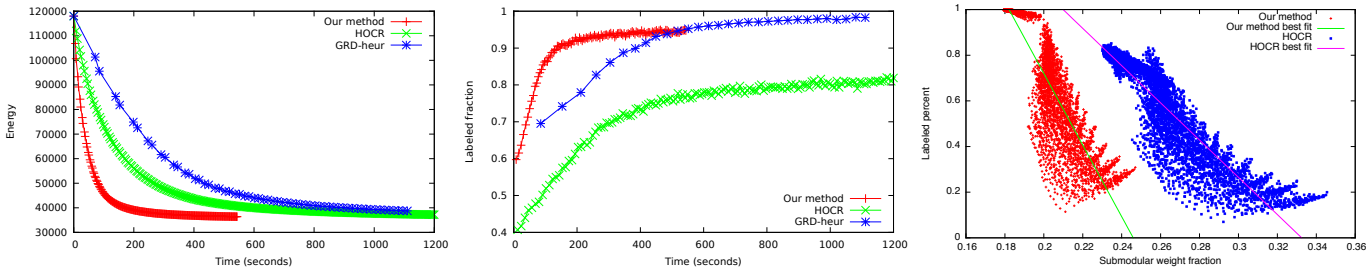

Fig. 4. Energy after each fusion move (left), and percentage of pixels labeled by QPBO (center), for the image at top of figure 3. Other images from [2] give very similar curves. (right) Fraction of pixels labeled by QPBO vs total weight of non-submodular edges (as a fraction of the total weight of all edges), along with best-fit lines for each method.

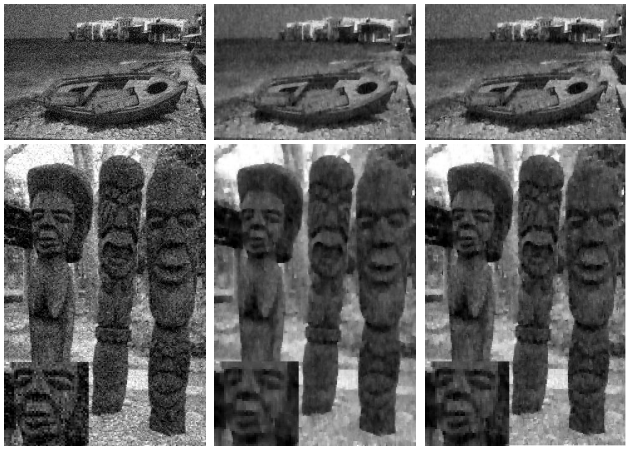

Fig. 3. Denoising examples. At left is the noisy input image, with our result in the middle and Ishikawa's at right. Results are shown after 30 iterations. More images are included in the supplemental material. To compare energy values with visual results, the images on the top row have energies 118,014, 26,103 and 38,304 respectively; those on the bottom have energies 118,391, 25,865 and 38,336.

methods have publicly available code implemented in C++, and provide very similar interfaces for setting up and optimizing a higher-order MRF.

For all experiments, we only report the results from the heuristic version of GRD, GRD-heur. Because the exact version solves an LP, running this method on vision-sized inputs proved to be prohibitive. A single iteration of fusion move took an average of an hour, compared to 40 seconds for GRD-heur. Consequently, it was impossible to run this method on the full dataset. Fortunately, GRD-heur has been shown [12], [13] to have similar optimization quality to the exact GRD, at the gain of significantly less computation time.

Our benchmark for evaluating the methods is the fields of experts priors for image denoising with fusion moves, using a dataset of 200 images, used in the original evaluation of HOCR [1], and later in the evaluation of GRD [12]. We used the same MRF, and as similar energy functions as possible. The fusion moves alternated between a randomly generated uniform image and a blurred image, and the energy function has clique size 4. We do multiple fusion moves on multiple images, so the effects of randomness are minimal.

To compare the effectiveness of each method on the individual binary subproblems, we ran all three algorithms on the first 30 fusion moves for each image, using the same starting point and proposal for each method (we averaged over only 30 iterations, because later iterations reduce the energy by much smaller amounts, and wash-out the differences in the methods). The results, averaged over the $200 \times 30 = 6000$ fusion moves are summarized in figure 6. Overall, we see that our method is strictly preferable to HOCR in all metrics, giving a better energy improvement per fusion move, labeling more pixels in QPBO, and taking less time. Compared to GRD, we see that GRD does label more pixels, and reduces the energy by an additional 35%; however, it takes over 7 times as long per iteration to compute.

To compare the effect of total-weight of non-submodular edges, we plotted the fraction of pixels labeled by QPBO vs. the fraction of non-submodular weight (divided by the total weight of all edges) in figure 4. There is a clear negative relation between these quantities — best fit lines had slopes of -15.6 and -8.2 for our reduction and HOCR respectively. Correspondingly, our method had a lower average fraction of non-submodular weight, 19.7% vs 26.0%, and a higher fraction labeled by QPBO, 76.1% vs 59.4%.

We also tested the effect of choosing which order to reduce variables. The default for all experiments was to reduce the variables in order $x_1, \ldots, x_n$. An alternative would be to choose the variables in decreasing order of number of positive terms. As predicted by our theoretical analysis, this difference had no measurable benefit. The "smart ordering" had 1% more average energy reduction, 0.4% fewer variables labeled, and won against the standard ordering (in terms of energy reduction) in only 42% of the 6000 fusion-moves. We also tested a random ordering, which performed somewhat worse than the standard ordering (better in only 2% of fusion moves, and 2% worse energy reduction). Because of extra bookkeeping, the smart ordering took 51% longer on average — consequently, we recommend using the standard order over more complicated schemes.

	Energy improvement	Percent labeled by QPBO	Time (seconds)
HOCR	1,302 (−45%)	59.4% (−22%)	14.1 (+150%)
GRD-heur	3,183 (+35%)	86.3% (+13%)	40.2 (+620%)
Our method	2,351	76.1%	5.6

Fig. 6. Performance comparison of reductions, on benchmarks in [2], averaged over 30 iterations of fusion move. Relative performance compared to our method in parenthesis.

	Extra variables	Non-submodular terms	Total terms
HOCR	224,346	421,897	1,133,811
Our method	236,806 (+6%)	38,343 (−90%)	677,183 (−40%)

Fig. 7. Total size of reductions, on Ishikawa's benchmarks in [2]. Relative performance of our method in parenthesis.

	Final energy	Time (seconds)
HOCR	32,199 (+2.3%)	2,050 (+102%)
GRD-heur	31,375 (−0.3%)	5,587 (+450%)
Our method	31,473	1,012

Fig. 5. Comparison of end-to-end performance on benchmarks in [2] at convergence of the fusion move optimization, averaged over all images. Relative performance compared to our method is shown in parenthesis.

As a second experiment, we compared the end-to-end performance of using each optimizer all the way through a complete run of the fusion-move algorithm. These results are displayed in figures 4 and 5.[2] Our method converges much faster, despite GRD reducing the energy by slightly more each step. Overall, the computational inefficiency of GRD greatly outweighs the marginal improvement in per-subproblem solution quality for fusion move.

The sizes of the obtained reductions for our method and the other term-rewriting reduction, HOCR, are summarized in figure 7. Overall, our method does better in practice than the asymptotic analysis in figure 1 suggests. As predicted, we produce many fewer non-submodular terms, but we also produce fewer submodular terms (a relative improvement of 10%).

Visual results are shown in figure 3. In the boat image our results appear more accurate in smooth areas like the water, and the face image (shown magnified at bottom left) is also noticably smoother. These results are after 30 fusion moves. The images after convergence (shown, along with more examples, in the supplemental material) are visually similar, though we still obtain lower energy.

Finally, we experimentally computed the local completeness of two early vision problems that are quite far from denoising, namely stereo [4] (clique size 3) and segmentation [5] (clique size 4). We analyzed the binary energy functions produced from 60 iterations of [4]. These energy functions have a very high local completeness; on average the energy functions are c-complete for $c = .98$, and their least locally complete energy function had $c = .96$. We also discovered that the higher-order segmentation energy function of [5] is absolutely locally complete ($c = 1$). These results suggest that our method may be particularly well suited to a number of important vision problems.

Acknowledgements: This work was supported by NSF grants IIS-0803705 and IIS-1161476/1161860, and by a joint CAPES (Brazil)/Fulbright (USA) fellowship to the second author under BEX-2387050/15061676. We thank Joyce Chen and Josh Schwartz for helpful comments.

2. We averaged together pairs of consecutive fusion moves in the graph shown at right in figure 4. This avoids the distracting sawtooth pattern visible in [1], [2], due to the alternation between random fusion moves and blurred fusion moves.

References

[1] H. Ishikawa, "Higher-order clique reduction in binary graph cut," in *CVPR*, 2009. 1, 2, 3, 6, 7, 8

[2] ——, "Transformation of general binary MRF minimization to the first order case," *TPAMI*, vol. 33, no. 6, 2010. 1, 3, 6, 7, 8

[3] V. Kolmogorov and C. Rother, "Minimizing nonsubmodular functions with graph cuts-a review," *TPAMI*, vol. 29, no. 7, pp. 1274–1279, July 2007, earlier version appears as technical report MSR-TR-2006-100. 1, 2

[4] O. Woodford, P. Torr, I. Reid, and A. Fitzgibbon, "Global stereo reconstruction under second-order smoothness priors," *TPAMI*, vol. 31, pp. 2115–2128, 2009. 1, 5, 8

[5] B. Andres, J. H. Kappes, U. Köthe, C. Schnörr, and F. A. Hamprecht, "An empirical comparison of inference algorithms for graphical models with higher order factors using opengm," in *DAGM-Symposium*, 2010, pp. 353–362. 1, 8

[6] A. Fix, A. Gruber, E. Boros, and R. Zabih, "A graph cut algorithm for higher-order Markov Random Fields," in *ICCV*, 2011. 1

[7] S. Roth and M. Black, "Fields of experts," *IJCV*, vol. 82, pp. 205–229, 2009. 1, 2, 5, 6

[8] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother, "A comparative study of energy minimization methods for Markov Random Fields," *TPAMI*, vol. 30, no. 6, pp. 1068–1080, 2008. 1, 2, 3

[9] J. H. Kappes, B. Andres, F. A. Hamprecht, C. Schnorr, S. Nowozin, D. Batra, S. Kim, B. X. Kausler, J. Lellmann, N. Komodakis *et al.*, "A comparative study of modern inference techniques for discrete energy minimization problems," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE, 2013, pp. 1328–1335. 1, 2

[10] P. Kohli, M. P. Kumar, and P. H. Torr, "P3 and beyond: Move making algorithms for solving higher order functions," *TPAMI*, vol. 31, no. 9, pp. 1645–1656, 2008. 1, 2

[11] X. Lan, S. Roth, D. Huttenlocher, and M. J. Black, "Efficient belief propagation with learned higher-order Markov Random Fields," in *ECCV*, 2006, pp. 269–282. 1, 2

[12] F. Kahl and P. Strandmark, "Generalized roof duality for pseudo-boolean optimization," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, 2011, pp. 255–262. 1, 2, 6, 7

[13] ——, "Generalized roof duality," *Discrete Applied Mathematics*, vol. 160, no. 1617, pp. 2419 – 2434, 2012. 1, 3, 6, 7
[14] D. Freedman and P. Drineas, "Energy minimization via graph cuts: Settling what is possible," in *CVPR*, 2005. 1, 2, 3
[15] V. Kolmogorov and R. Zabih, "What energy functions can be minimized via graph cuts?" *TPAMI*, vol. 26, no. 2, pp. 147–59, 2004. 1, 2, 3
[16] I. Rosenberg, "Reduction of bivalent maximization to the quadratic case," Centre d'Etudes de Recherche Oprationnelle, Tech. Rep., 1975. 1, 2, 3
[17] C. Rother, P. Kohli, W. Feng, and J. Jia, "Minimizing sparse higher order energy functions of discrete variables," in *CVPR*, 2009, pp. 1382–1389. 1, 3
[18] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *TPAMI*, vol. 23, no. 11, pp. 1222–1239, 2001. 2
[19] P. Hammer and S. Rudeanu, *Boolean Methods in Operations Research and Related Areas*. Springer, 1968. 2
[20] E. Boros and P. L. Hammer, "Pseudo-boolean optimization," *Discrete Applied Mathematics*, vol. 123, no. 1-3, 2002. 2
[21] V. Lempitsky, C. Rother, S. Roth, and A. Blake, "Fusion moves for Markov Random Field optimization," *TPAMI*, vol. 32, no. 8, pp. 1392–1405, Aug 2010. 2, 6
[22] P. Hammer, "Some network flow problems solved with pseudo-boolean programming," *Operations Research*, vol. 13, pp. 388–399, 1965. 2
[23] P. L. Hammer, P. Hansen, and B. Simeone, "Roof duality, complementation and persistency in quadratic 0-1 optimization," *Mathematical Programming*, vol. 28, pp. 121–155, 1984. 2
[24] N. Komodakis and N. Paragios, "Beyond pairwise energies: Efficient optimization for higher-order MRFs," in *CVPR*, 2009, pp. 2985–2992. 2
[25] S. Zivny, D. Cohen, and P. Jeavons, "The expressive power of binary submodular functions," in *Mathematical Foundations of Computer Science 2009*, ser. LNCS, 2009, vol. 5734, pp. 744–757. 2, 3, 4
[26] C. Wang, N. Komodakis, and N. Paragios, "Markov random field modeling, inference & learning in computer vision & image understanding: A survey," *Computer Vision and Image Understanding*, vol. 117, no. 11, pp. 1610 – 1627, 2013. 3
[27] D. Schlesinger, "Exact solution of permuted submodular minsum problems," in *Energy Minimization Methods in Computer Vision and Pattern Recognition*, 2007, pp. 28–38. 3