Approximate inference using unimodular graphs in dual decomposition

Chetan Bhole
Dept. of Computer Science, Univ. of Rochester
bhole@cs.rochester.edu

Justin Domke
Machine Learning Group, NICTA
Justin.Domke@nicta.com.au

Daniel Gildea
Dept. of Computer Science, Univ. of Rochester
gildea@cs.rochester.edu

Abstract

We use the property of unimodular functions to perform approximate inference with dual decomposition in binary labeled graphs. Exact inference is possible for a subclass of binary labeled graphs that have unimodular functions. We call such graphs unimodular graphs. These are graphs where the submodular and non-submodular edges follow a specific pattern– essentially that an isomorphism or "flipping" exists to a fully submodular graph. Examples of unimodular graphs include tree-structured graphs, submodular graphs, and bipartite graphs with all non-submodular edges. We investigate the use of unimodular graphs in dual decomposition, based on different decomposition strategies. Experimentally, for image segmentation problems, we find that decomposition using unimodular graphs outperform traditional tree-based dual decomposition. Dual decomposition is also more easily parallelizable.

1 Introduction

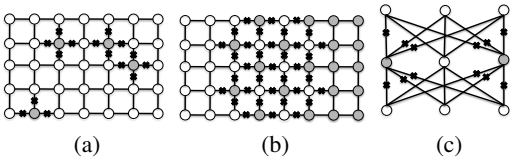
This paper considers the common Maximum a Posteriori (MAP) problem of minimizing an energy function of the form $E(\mathbf{x}) = \sum_{(i,j)\in\mathcal{E}} e(x_i, x_j) + \sum_{i\in\mathcal{V}} e(x_i)$ over binary vectors $\mathbf{x}$, where $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is an undirected graph where $\mathcal{V}$ and $\mathcal{E}$ are the vertices and edges of the graph respectively. As this problem is NP-hard in general, research has focused in two directions. The first is finding special cases where the energy can be minimized efficiently i.e. in polynomial time. Common examples of this include the case where $\mathcal{G}$ is tree-structured using dynamic programming [2], planar problems with no univariate energies [12], binary labeled graphs with single cycle hidden variables that can have univariate energies [1], in binary labeled graphs when the interaction energies $e(x_i, x_j)$ are submodular, or in lesser known unimodular functions [6]. The second direction is finding approximate algorithms for minimizing general energies, such as TRW-S [7], max-product loopy belief propagation [11], dual decomposition [9], Quadratic Pseudo Boolean Optimization (QPBO) [8] or inference using planar graphs [5]. These algorithms may even provably find the exact solution for individual problem instances, even in the absence of a general guarantee.

In this paper, we propose the use of unimodular functions instead of trees or submodular functions in dual decomposition. The motivation of using larger subgraphs compared to trees is to improve convergence speed. The contributions of this paper include study of dual decomposition using subproblems of unimodular energies, proposing two alternatives for subgraph construction, one from building on spanning trees and the other by splitting the graph into submodular and non-submodular subgraphs, and present experimental results on synthetic

problems with different energy patterns and a real image segmentation problem. We find that our setup greatly improves the convergence of dual decomposition over the use of trees.

2 Unimodular graphs

An edge (i, j) is said to be **submodular** if $e_{ij}(0,0) + e_{ij}(1,1) \leq e_{ij}(0,1) + e_{ij}(1,0)$ and otherwise non-submodular. We will also refer to energy E as submodular if this is true for all edges of the graph. For such energies, it is possible to solve the problem efficiently by reducing it to a max-flow/min-cut problem [3]. Whether or not an energy is unimodular depends on global properties of the energy and graph structure, and so cannot be characterized as easily as submodularity.



(a) (b) (c)

Figure 1: Examples of unimodular graphs. Edges marked with a "$\times$" are non-submodular, while all others are submodular. In each case, if the shaded nodes are flipped, a submodular energy is obtained.

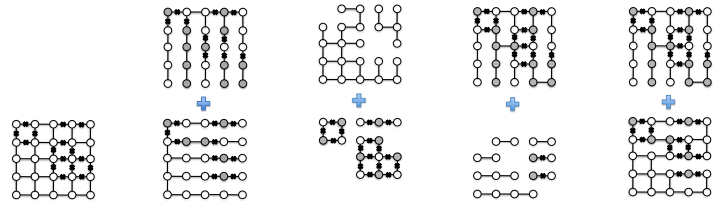
Definition 1. Given a subset $F \subseteq \mathcal{V}$, flipped energy functions ("relabeled") are given by $E^F(\mathbf{x}) = \sum_{(i,j)\in\mathcal{E}} e_{ij}(x'_i, x'_j) + \sum_{i\in\mathcal{V}} e_i(x'_i)$ where $x'_i = x_i$ if $i \notin F$, or $x'_i = 1 - x_i$ if $i \in F$.

Lemma 2. *The submodularity of an edge energy $e_{ij}(x_i, x_j)$ will be changed (from submodular to non-submodular or vice-versa) in $e_{ij}(x'_i, x'_j)$ if exactly one of i and j is in F. Otherwise it will remain the same. (Proof in supplemental text.)*

Theorem 3. *E^F will be submodular if and only if: For each non-submodular edge (i, j) exactly one of i and j is in F and for each submodular edge (i, j) either (a) both i and j are in F or (b) neither i nor j is in F.*

An energy function is **unimodular** if there exists a subset $F \subseteq V$ such that $E^F(\mathbf{x})$ is submodular. We call graphs that have unimodular functions as unimodular graphs. There exists an isomorphism between a unimodular energy and the corresponding $E^F(\mathbf{x})$ submodular energy. Graph cuts will give a minimum energy labeling for the flipped graph (corresponding to $E^F(\mathbf{x})$). After re-flipping the nodes back from the flipped graph labeling, a minimum energy labeling is obtained for the original unimodular graph. Fig. 1 shows some examples of unimodular graphs. One can imagine flipping the shaded nodes to obtain a submodular graph. An efficient method (complexity $\mathcal{O}(|\mathcal{E}| + |\mathcal{V}|)$) to check if an energy is unimodular, based on a reduction to 2-SAT, has been proposed [6]. In our implementation, we use a breath first search algorithm instead. QPBOP [8] implements a variant of the construction of Hammer et al. [6] that performs inference on general graphs but can leave nodes unlabeled. It however solves unimodular functions efficiently. In our algorithm (detailed differences in supplemental text), we only wish to use unimodular graphs for exact inference, as a dual decomposition subproblem. Thus, we explicitly verify that the subproblem is unimodular by finding a set F (using $O(n)$ breadth first search) that renders it submodular, then explicitly construct the modified energy keeping the graph size the same, and solve it via graph cuts.

3 Dual Decomposition using Unimodular Graphs



(a) Original Graph (b) DD-T (c) DD-S (d) DD-F (e) edge sharing

Figure 2: Methods for splitting a graph into subgraphs. (a) A binary graph with non-submodular edges marked with crosses. Flipped nodes are darkened. (b) DD-T splits into two spanning trees. (c) DD-S splits into one subgraph for all submodular edges, and one for all non-submodular. (d) DD-F splits into one subgraph being a tree with edges added greedily, and one graph with the remaining edges. (e) Decomposition into two subgraphs initialized from spanning trees shown in (b).

Komodakis et al. [9], observed that, by use of the classic optimization strategy of *dual decomposition*, minimization of the energy function can be approximated by selecting a set of subgraphs G (so

that each vertex and edge in the original graph is present in at least one subgraph) and solving the alternative problem

$$\max_{\{e_i^G\}} \min_{\{x^G\}} \sum_G E_G(x^G), \quad \text{where } E_G(x^G) = \sum_{i \in \mathcal{V}} e_i^G(x_i^G) + \sum_{(i,j) \in G} e_{ij}^G(x_i^G, x_j^G) \quad \text{s.t. } e_i(x_i) = \sum_G e_i^G(x_i) \tag{1}$$

where $\{e_{ij}^G\}$ is implicitly given by some (fixed) set of functions such that $e_{ij}(x_i, x_j) = \sum_{G:(i,j)\in G} e_{ij}^G(x_i, x_j)$ and where $E(x) = \sum_G E_G(x^G)$. We will not need to update e_{ij}^G the way we decompose our graphs. The motivation for this is that the graphs G should be chosen in such a way that the subproblems $\min_{x^G} E_G(x^G)$ can be solved quickly. The most common subproblems are tree-structured problems which can be solved efficiently by dynamic programming. Fig. 2(b) shows an example of using trees as subproblems. Komodakis et al. [9, section 6.3] also discuss high-treewidth graphs where the edge costs e_{ij}^G are submodular and so can be solved via graph cuts. The outer problem is maximizing $D(\{e_i^G\}) = \min_{\{x^G\}} \sum_G E_G(x^G)$ over the constraint set induced by Eq. 1. D can be shown to be continuous, concave and non-differentiable. A supergradient can be computed by $\frac{dD}{de_i^G(a)} = I[x_i^G = a]$ where $x^G = \arg\min_{x^G} E_G(x^G)$. D can be maximized via projected supergradient ascent.

Dual decomposition can also be used when subgraphs are not all spanning. Then, univariate energies e_i^G are only adjusted for variables that are shared between at least two subgraphs. When some vertices or edges are unique to only one subgraph, no updates are necessary for those vertices and edges. It is also useful to point out that an important advantage of dual decomposition methods over other methods like QPBO is that the decomposition methods can be used to parallelize inference in very large graphs that might not fit on one machine and thus can be decomposed into smaller subgraphs and run on multiple machines. We make use of unimodular graphs as subproblems and it is easy to see that any submodular or tree-structured [8] graph will be a unimodular graph, and so this is a generalization of previously-used subproblems for dual decomposition.

We consider two methods for decomposing a graph into unimodular graphs. The first, DD-submodular (DD-S), is illustrated in Fig. 2(c). Given a grid graph, it is split into one subgraph consisting of all submodular edges, and one subgraph containing all non-submodular energies so each graph is a valid unimodular graph. (We use the property that for a bipartite graph consisting of all non-submodular energies a flipping of alternate nodes will always render it submodular.). It is also important to note that more general non-grid graphs can also be decomposed using this strategy except that there will possibly be more than two subgraphs depending on the graph structure. The second method, DD-flipper (DD-F) is illustrated in Fig. 2 (d) and (e). Here, we begin with a decomposition consisting of spanning trees, as might be used in traditional dual decomposition. After this we try to add edges to the spanning trees so that the graph being constructed is an unimodular graph. It is always possible to convert a tree to a submodular tree by flipping some of the nodes. This determines the status of each node flipped or unflipped. Next, each edge from the original graph is considered to be added according to the conditions in Theorem 3. If it can be added (can be checked in constant time) to the decomposition, it is, otherwise it is left out. Thus, one ends up with a decomposition with subproblems containing at least as many edges as in a tree decomposition, and possibly many more. Potentially, we can avoid sharing edges between subgraphs as in Fig. 2(d) (by starting with one spanning tree and a forest of smaller trees so that each edge exists in one of the trees) or allow edge sharing (as in Fig. 2(e)) by dividing the pairwise energies by the number of subgraphs and making no updates to these energies during dual decomposition. In both cases, we only need two subgraphs for rectangular grid graphs. This is because in the first case where there is no edge sharing, removing a spanning tree from a rectangular grid leaves behind a forest of trees that are always unimodular. In the edge sharing case, the graph is broken into spanning trees as in Fig. 2 (b) and so both trees already cover all edges of the graph. Edges are added to these trees to make them more connected while maintaining the unimodular property.

As discussed by Komodakis et al. [9, Section 6], any decomposition using subproblems with the property of *integrality* will achieve a dual objective with the same value (namely that of a LP-relaxation of the original integer program). Tree-structured and submodular graphs are both known to obey integrality. Similarly, since unimodular graphs are essentially a relabeling of a submodular graph, they will also obey this property, and so decomposition based on unimodular graphs will obtain the same dual. Nevertheless, certain decompositions can be easier to round into a higher-quality primal solution with fewer iterations, and convergence rates can differ. Both of these are observed in our experiments.

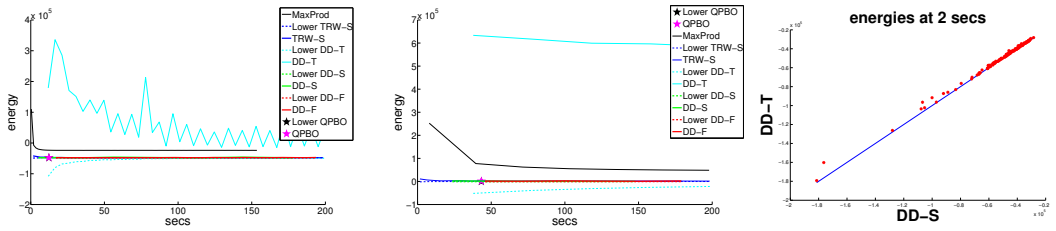


Figure 3: Inference using different algorithms on a 1000 x 1000 grid (left and middle). Details of the graph structures in text. Best viewed in color. DD-S, DD-F and TRW-S are along the flat lines with minimum energies but magnification shows DD-S does better than TRW-S.

Table 1: Segmentation metrics for 128 test images

	TRW-S	DD-T	DD-F	DD-S	QPBO	MaxProd
% pixel error	8.97	10.12	9.44	8.99	**8.86**	18.10
avg precision	0.8735	0.7857	0.8744	**0.8772**	0.8687	0.8278
avg recall	0.7606	0.7002	0.7515	0.7596	**0.7722**	0.3281

4 Experiments and Conclusions

We found edge sharing for both DD-T and DD-F perform no better than without edge sharing and have the additional penalty of running inference on more than one larger subgraph. Hence, unless explicitly mentioned we provide results for the case where no edge sharing is done. We obtain primal solutions for the dual decomposition methods as follows. For DD-T and DD-F we take the best primal energy from the larger subgraph containing all nodes. For DD-S, we use the submodular graph for shared nodes, and otherwise whichever graph contains a node. We use the adaptive step size method for projected subgradient [9].

Synthetic experiments : All the synthetic experiments are based on 1000×1000 node grids with randomly chosen energy terms. All algorithms are implemented in C++ and use a single core of a 3.33GHz processor, based on and extending a MRF package [13]. We generate the local data cost terms as $e(x_i) \sim \mathcal{N}(0, 1)$. For submodular pairwise terms, we set $e(x_i, x_j) = 0$ when $x_i = x_j$ and take $e(x_j, x_j) \sim |\mathcal{N}(0, \sigma)|$ when $x_i \neq x_j$. The experiments we show are for $\sigma = 3$. Changing this parameter did not seem to have effect on the results. For non-submodular edges, we do the opposite– set $e(x_i, x_j) = 0$ when $x_i \neq x_j$ and otherwise take the absolute value of a Normal variable. Our termination criteria for algorithms that did not converge was 500 iterations.

We consider the results on graphs that are not unimodular. (More experiments and plots in supplemental text.) Fig. 3 left shows results from a graph which has the left half edges with submodular energies and the remaining edges (right half and edges connecting left and right halves) with nonsubmodular energies. Both DD-F and DD-S perform similar to TRW-S and QPBO and better than DD-T. Even though, it is not the global optimum, QPBO does provide its solution quickly. Fig. 3 (right) is based on a graph emulating an image segmentation task. There exists a square boundary of size of 500x500 in the center with non-submodular edges. All other edges, both inside (the "foreground") and outside (the "background") the square are submodular. DD-T does worse than our decomposition methods in the experiments above.

Image Segmentation : We use the Weizman horse dataset for segmentation. A MRF model has been trained using 200 horse images. 128 images were reserved for testing. The two labels (two class MAP estimation) are pixels belonging to a horse and those belonging to the background. The parameters of the model were trained using truncated fitting using TRW-S based on the univariate logistic loss [4], [10]. Univariate terms were predicted using a linear function of 100 local features modeling color, position and Histogram of Oriented Gradients (HOG). Pairwise terms were predicted using a linear function of 42 features modeling discretized gradients between neighboring pixels. The same trained parameters were used during testing inference by all the inference methods. Table 1 provides the total percentage pixel error values on the test images, i.e., all foreground and background image pixels are considered to compute pixel error. Energy plot (Fig. 3 right) shows DD-S does better than DD-T on each test image.

Conclusion : We have discussed how to construct unimodular graphs and shown that unimodular graphs are better than the traditional trees for dual decomposition since they greatly improve convergence speed. Experimentally, the best results are obtained by using a decomposition with unimodular graphs with fewer shared nodes between subgraphs.

References

[1] S. M. Aji, G. B. Horn, and R. J. McEliece. On the convergence of iterative decoding on graphs with a single cycle. In *IEEE International Symposium of Information Theory*, page 276, 1998.

[2] Christopher Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[3] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:2001, 2001.

[4] J. Domke. Parameter learning with truncated message-passing. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2937–2943, 2011.

[5] A. Globerson and T. Jaakkola. Approximate inference using planar graph decomposition. In *Neural Information Processing Systems*, pages 473–480, 2006.

[6] P.L. Hammer, P. Hansen, and B. Simeone. Roof duality, complementation and persistency in quadratic 0-1 optimization. *Math. Programming*, 28:121–155, 1984.

[7] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1568–1583, October 2006.

[8] V. Kolmogorov and C. Rother. Minimizing nonsubmodular functions with graph cuts - a review. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 29(7):1274–1279, 2007.

[9] N. Komodakis, N. Paragios, and G. Tziritas. MRF energy minimization and beyond via dual decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):531 –552, march 2011.

[10] Talya Meltzer, Amir Globerson, and Yair Weiss. Convergent message passing algorithms - a unifying view. In *Conference on Uncertainty in Artificial Intelligence*, 2009.

[11] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

[12] N. Schraudolph and D. Kamenetsky. Efficient exact inference in planar Ising models. In *Neural Information Processing Systems*, 2009.

[13] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, A. Agarwala, and C. Rother. A comparative study of energy minimization methods for Markov random fields. In *European Conference on Computer Vision*, pages 16–29, 2006.