An Empirical Comparison of Inference Algorithms for Graphical Models with Higher Order Factors Using OpenGM

Björn Andres, Jörg H. Kappes, Ullrich Köthe,
Christoph Schnörr, and Fred A. Hamprecht

HCI, IWR, University of Heidelberg

Abstract. Graphical models with higher order factors are an important tool for pattern recognition that has recently attracted considerable attention. Inference based on such models is challenging both from the view point of software design and optimization theory. In this article, we use the new C++ template library OpenGM to empirically compare inference algorithms on a set of synthetic and real-world graphical models with higher order factors that are used in computer vision. While inference algorithms have been studied intensively for graphical models with second order factors, an empirical comparison for higher order models has so far been missing. This article presents a first set of experiments that intends to fill this gap.

1 Introduction and Related Work

Graphical models have been used very successfully in pattern analysis, usually as probabilistic models in which the graph expresses conditional independence relations on a set of random variables [1,2]. Graphical models are not restricted to probabilistic modeling and have also been used more generally to represent the factorization of arbitrary multivariate functions (w.r.t. a given operation) into factors that depend on subsets of all variables [3]. Factor graphs [3,4] are a common way of representing this factorization. In this article, we focus on functions which depend on discrete variables that can attain only finitely many values.

The optimization of such functions is important to perform MAP estimation (inference) under the assumption of a probabilistic model as well as to find configurations with minimal energy in non-probabilistic models. In several special cases the optimization problem can be solved in polynomial time (in the number of variables), in particular if the graphical model is acyclic [1] or if the energy function is (permutation) submodular [5,6,7]. However, if the model does not belong to either of these classes, exact optimization is NP-hard in the general [6].

For several years, research efforts focused mainly on functions which decompose into factors that depend on at most two variables. However, in order to model more complex problems, higher order factors have recently become an active field of research [8,9,10,11]. While second order factors are conceptually simple, higher order factors are challenging from the view point of optimization theory and implementation.

M. Goesele et al. (Eds.): DAGM 2010, LNCS 6376, pp. 353–362, 2010.

At least two types of algorithms exist : *message passing algorithms* (e.g. Loopy Belief Propagation (LBP) [1,12,13] and Tree-Reweighted Belief Propagation (TRBP) [14,15]) and *search-based algorithms* (e.g. A* search [16], Iterated Conditional Modes (ICM) [17], and a generalization if ICM called the Lazy Flipper [18]). While message passing algorithms implicitly approximate the objective function, search-based algorithms either restrict the search space or else have a runtime that is in the worst case exponentially large in the number of variables. All these algorithms output upper bounds on the global minimum (when used for minimization). TRBP in addition affords a lower bound on the global minimum.

For the sake of comparability, it is essential that these different algorithms are used with the same underlying data structures. While data structures for factors that depend only on binary variables are simple, consisting mostly of low-level functions close to machine code, similar data structures which allow the variables to have different domains are non-trivial and require unavoidable overhead. To quantify the effect from using different data structures on the absolute runtime of a particular algorithm, the same algorithm needs to be run with different underlying data structures.

We developed the extendible C++ template library OpenGM for this comparison. OpenGM allows the programmer to use different inference algorithms and different factor data structures interchangeably. While the experiments in this article are restricted to graphical models with discrete variables (binary and non-binary), OpenGM is general enough to also work with parameterized factors that depend on continuous variables. In contrast to Infer.NET [19] which has been published without source code under a fairly restrictive licence and libDAI [20] which is open source and published under a General Public Licence that forces the user to publish derived code under the same licence (so-called copyleft terms), the OpenGM source code will be published under the MIT licence that imposes almost no restrictions and does not contain copyleft terms.

We show in this article that the examined algorithms perform very differently, depending on the structure of the graphical model, both in terms of the upper bound on the global minimum as well as in terms of absolute runtime. Moreover, we show that the absolute runtime crucially depends on the data structures that are used to represent factors. In contrast to the Middleburry MRF Benchmark [21] and the quantitative comparison in [22], our comparison includes graphical models with higher order factors, different topologies, and different variable domains.

2 Optimization Problem and Algorithms

The models considered in this article are given in factor graph notation. A factor graph $G = (V, F, E)$ is a bipartite graph that consists of three sets: a set of variable nodes V, a set of factor nodes F, and a set of edges $E \subseteq V \times F$ that connect variables to factors. For each factor node, $\mathrm{ne}(f) := \{a \in V : (a, f) \in E\}$ denotes the set of all variable nodes that are connected to the factor.

With each variable node $a \in V$, a variable $x_a \in \mathcal{X}_a$ is associated. For a set of variable nodes $A \subseteq V$, $\mathcal{X}_A = \bigotimes_{a \in A} \mathcal{X}_a$ denotes the Cartesian product

of the respective variable domains. Corresponding sequences of variables are written as $x_A = (x_a)_{a \in A}$. Moreover, with each factor node $f \in F$, a function $\varphi_f : \mathcal{X}_{\mathrm{ne}(f)} \to \mathbb{R}$ is associated. Together with some commutative and associative operation (addition throughout this article), the factor graph describes a function over $\mathcal{X} := \mathcal{X}_V$:

$$J(x) = \sum_{f \in F} \varphi_f(x_{\mathrm{ne}(f)}) \ . \tag{1}$$

We will consider the following discrete optimization problem:

$$x^* = \arg\min_{x \in \mathcal{X}} \sum_{f \in F} \varphi_f(x_{\mathrm{ne}(f)}) \tag{2}$$

where, for all $a \in V$, $\mathcal{X}_a = \{1, \ldots, L\}$.

Five algorithms to tackle this problem are compared in this article, the message passing algorithms LBP and TRBP as well as the search-based algorithm A* search, ICM, and the Lazy Flipper.

Motivated by Pearl's Belief Propagation algorithm [1] which is exact for tree structured models, LBP [12,13] is one of the toady's standard methods. It applies the message passing rules for acyclic graphs to graphs with cycles, without providing any convergence guarantees. Even if this method is rather heuristic, it shows quite good performance in real-world applications.

The more mathematically sound message passing algorithm TRBP was presented by Wainwright [14] who considers a convex relaxation of the problem by a tree-decomposition. The resulting message passing algorithm can be seen as a fixed point iteration on this convex outer relaxation. In subsequent work [15], Kolmogorov proposed a sequential version (TRW-S) which is guaranteed to converge to fix-points satisfying the so-called week tree agreement.

Very recently, several methods [16,23] have been suggested which solve discrete minimization problems in computer vision by branch and bound search. We implemented the methods presented in [16] which transform the optimization problem (2) into a shortest path problem on a graph whose size is exponential in the size of the graphical model. The shortest path problem is then solved by A* search using a tree-based approximation of the original graph to approximate the further cost while searching. This algorithm is guaranteed to converge to global optima and performs well on some classes of models. However, the worst-case runtime is exponential in the number of variables.

ICM [17] is a simple algorithm that keeps all variables except one fixed in each step and adjusts the free variable such that the objective function is minimized. The variables are visited repeatedly in a certain order until no alteration of a single variable can further reduce the value of the objective function.

The Lazy Flipper [18] is a generalization of ICM for models with binary variables. It extends the ICM search to subsets that contain more than one variable. The Lazy Flipper is more efficient than exhaustive search (by an amount that can be exponential in the number of variables) because the search space is restricted to a set of variables which are connected via factors in the graphical

model and which have not been visited in previous iterations. The maximum size of subsets can be supplied as a parameter to the algorithm. When set to infinity, the Lazy Flipper is guaranteed to converge to a global optimum. Its runtime is exponential in the worst case.

3 Empirical Comparison of Inference Algorithms

For a comparison of the algorithms, we consider both synthetic and real-world graphical models with discrete variables that appear in computer vision applications. Synthetic models have the advantage that several instances of the same model can be generated which allows us to decide whether differences in performance and runtime are significant.

3.1 Synthetic Models

We characterize models by their *graph structure*, the *number of variables*, the *number of values each variable can attain*, the *order of factors*, and the *distribution of factor values*. For simplicity, we let the number of values be equal for all variables and let each model contain only factors of the orders 1 and M. Two different graph structures are considered, each with two different orders:

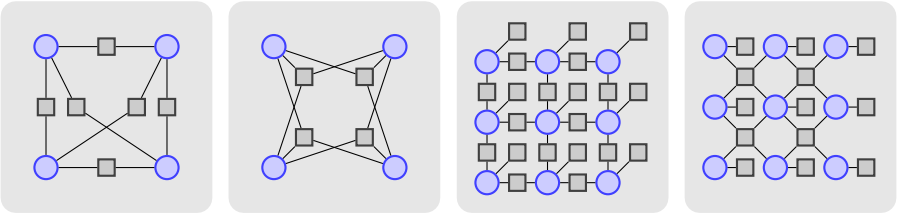
1. Fully connected models with second and third order factors, respectively, consisting of 8 variables each of which can attain 20 different values. These models are often used in part based object detection [16].
2. Grid graph models with first order factors as well as second and fourth order factors, respectively, which are frequently used in image segmentation. A grid with 40 times 40 (1600) variables is used, each variable attaining 2 and 5 different values, respectively. For the grid-based models, the additional parameter $\lambda \in [0, 1]$ controls the coupling strength between first and higher order factors according to

$$J(x) \quad = \quad (1-\lambda) \sum_{f \in F, |\mathrm{ne}(f)|=1} \varphi_f(x_{\mathrm{ne}(f)}) \quad + \quad \lambda \sum_{f \in F, |\mathrm{ne}(f)|>1} \varphi_f(x_{\mathrm{ne}(f)}) \tag{3}$$

The four different graph structures for the synthetic models are depicted in Fig. 3.1 for a small number of variables. The factors of all models are sampled independently, an assumption which may not hold in real data. Two different types of factors are considered: The values of *uniform factors* are sampled uniformly from the interval $[0, 1]$. For *log-uniform factors*, values v are sampled uniformly from the interval $(0, 1]$, and the values of factors set to $-\log(v)$. This results in very selective factors which in this form often appear within part based models [16,23]. For each model, an ensemble of 10 instances is generated.

3.2 Real-World Models

In addition, we consider 100 graphical models obtained from the 100 natural test images in the Berkeley Segmentation Dataset [24] that are used to remove



(a) $\mathrm{FULL}^2(4,\cdot)$ (b) $\mathrm{FULL}^3(4,\cdot)$ (c) $\mathrm{GRID}^2(9,\cdot)$ (d) $\mathrm{GRID}^4(9,\cdot)$

Fig. 1. Graph structures of the synthetic models with less variables

excessive boundaries from image over-segmentations. These real-world models have on average 8800 binary variables, 5700 third order and 100 fourth order factors. Each variable corresponds to a boundary between segments, indicating whether this boundary is to be removed (0) or preserved (1). Unary factors relate these variables to the image content while non-submodular third and fourth order factors connect adjacent boundaries, supporting the closedness and smooth continuation of preserved boundaries.

3.3 Results

In the tables below, the mean upper bounds $\mathbb{E}(J)$ on the minimum energy and the mean runtimes $\mathbb{E}(t)$ (in seconds) over these ensembles are shown. Runtimes are measured on one core of an Intel Pentium Dual Core at 2.00 GHz. In addition, we note in the tables how often each algorithm outputs the smallest upper bound on the minimum energy among the compared algorithms. When algorithms use a data structure that is specialized for binary variables, this is marked with a 2 at the end of the algorithm name.

Results for the fully connected synthetic models are shown in Tab. 1. While A* search guarantees global optimality, LBP, TRBP, and ICM show inferior

Table 1. Results for fully connected models with 8 variables and 20 labels: On the 2nd order model $\mathrm{FULL}^2(8,20)$ A* search outperforms LBP and TRBP in terms of energy and runtime. LBP sometimes also ends up in a global optimum for 2nd order models. On the 3rd order model $\mathrm{FULL}^3(8,20)$, A* search still calculates the global optima, contrary to all other algorithms

	2nd order						3rd order					
	uniform			log			uniform			log		
	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best
LBP	8.21	0.88	40%	12.70	1.06	20%	24.14	197.69	0%	51.69	197.38	0%
TRBP	10.19	1.54	0%	21.36	1.58	0%	25.55	99.00	0%	55.52	76.63	0%
ASTAR	4.82	0.82	**100%**	6.12	0.55	**100%**	14.19	20693	**100%**	20.05	16881	**100%**
ICM	10.37	0.00	0%	20.29	0.00	0%	22.58	0.00	0%	40.45	0.00	0%

performance. The Lazy Flipper cannot be applied because these models have non-binary variables.

Results for the synthetic grid models are shown in Tab. 2–5. The Lazy Flipper consistently outperforms LBP, TRBP, and ICM in terms of quality. Surprisingly, LBP performs overall better than TRBP. While the use of an optimized data structure for binary variables results in a speed-up factor of 2 for LBP, TRBP, and ICM, the change is marginal for Lazy Flipper because the Lazy Flipper spends most of the time on graph traversal, while LBP and TRBP sum up and minimize multi-dimensional factors during message passing and ICM mostly evaluates factors for certain assignments of values to the variables. It is hard to make any general claims on the runtime of LBP and TRBP since they do not convergence, and so the runtime usually depends linearly on the maximal number of iterations.

Table 2. Results for the binary second order grid model $\mathrm{GRID}^2(1600, 2)$ consisting of 40 times 40 variables. For smaller coupling strength $\lambda = 0.25$, LBP and TRBP perform comparable with the Lazy Flipper. For larger coupling strength, the Lazy Flipper consistently outperforms LBP and TRBP in comparable runtime.

	uniform					
	$\lambda = 0.25$			$\lambda = 0.75$		
	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best
LBP	754.76	7.30	**90%**	995.65	31.60	0%
TRBP	754.78	18.44	**80%**	995.62	39.65	0%
LF	754.76	10.36	**90%**	993.68	10.77	**100%**
ICM	790.51	2.67	0%	1303.53	2.77	0%
LBP2	754.76	3.40	**90%**	995.65	14.58	0%
TRBP2	754.78	8.24	**80%**	995.62	17.66	0%
LF2	754.76	10.00	**90%**	993.68	10.56	**100%**
ICM2	790.51	1.37	0%	1303.53	1.42	0%

	log					
	$\lambda = 0.25$			$\lambda = 0.75$		
	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best
LBP	1229.87	17.04	30%	1599.70	31.35	0%
TRBP	1229.89	23.58	20%	1594.76	39.74	0%
LF	1229.60	10.31	**90%**	1588.69	10.87	**100%**
ICM	1379.28	2.70	0%	2542.86	2.82	0%
LBP2	1229.87	7.90	30%	1599.70	14.48	0%
TRBP2	1229.89	10.47	20%	1594.76	17.54	0%
LF2	1229.60	10.03	**90%**	1588.69	10.61	**100%**
ICM2	1379.28	1.38	0%	2542.86	1.46	0%

Table 3. Results for the binary 4th order grid model $\mathrm{GRID}^4(1600, 2)$ consisting of 40 times 40 variables. Compared to the 2nd order model, the energy functions are much more challenging which results in a 17-times longer runtime for the Lazy Flipper. Similar to the 2nd order model, the Lazy Flipper outperforms LBP and TRBP, even for the weakly coupled models.

	uniform					
	$\lambda = 0.25$			$\lambda = 0.75$		
	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best
LBP	562.19	142.68	0%	497.22	142.90	0%
TRBP	570.41	161.19	0%	555.47	142.37	0%
LF	558.44	170.07	**100%**	449.93	222.45	**100%**
ICM	589.86	1.85	0%	700.53	1.85	0%
LBP2	562.19	64.55	0%	497.22	64.68	0%
TRBP2	570.41	71.60	0%	555.47	62.63	0%
LF2	558.44	167.28	**100%**	449.93	217.22	**100%**
ICM2	589.86	1.01	0%	700.53	1.03	0%

	log					
	$\lambda = 0.25$			$\lambda = 0.75$		
	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best
LBP	874.94	142.77	0%	786.36	142.00	0%
TRBP	891.07	161.69	0%	888.04	163.44	0%
LF	863.72	170.50	**100%**	683.87	238.41	**100%**
ICM	975.74	1.88	0%	1339.57	1.93	0%
LBP2	874.94	64.68	0%	786.36	64.08	0%
TRBP2	891.07	71.77	0%	888.04	71.98	0%
LF2	863.72	164.88	**100%**	683.87	231.82	**100%**
ICM2	975.74	1.01	0%	1339.57	1.05	0%

Table 4. Results for the 2nd order grid models with 5 labels, $\mathrm{GRID}^2(1600, 5)$. For this non-binary problem, we compare LBP, TRBP and ICM. For small coupling strength, LBP and TRBP perform comparable; LBP shows significantly better performance for larger coupling strength

	uniform						—log—					
	$\lambda = 0.25$			$\lambda = 0.75$			$\lambda = 0.25$			$\lambda = 0.75$		
	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best
LBP	514.55	10.71	10%	706.35	10.75	**100%**	710.62	10.70	60%	1035.78	10.87	**100%**
TRBP	514.22	13.70	90%	772.77	13.71	0%	711.08	13.73	40%	1164.19	13.67	0%
ICM	591.13	8.27	0%	1235.02	8.30	0%	1024.98	8.22	0%	2426.99	8.58	0%

Table 5. Results for the 4th order grid models with 5 labels, $\mathrm{GRID}^4(1600, 5)$. LBP outperforms TRBP when both run the same number of iterations.

	uniform					
	$\lambda = 0.25$			$\lambda = 0.75$		
	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best
LBP	344.87	271.34	**100%**	354.92	271.24	**90%**
TRBP	414.42	284.81	0%	368.98	284.87	10%
ICM	390.75	5.96	0%	636.23	5.96	0%

	log					
	$\lambda = 0.25$			$\lambda = 0.75$		
	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best
LBP	452.70	271.48	**100%**	467.57	271.61	**100%**
TRBP	547.41	285.03	0%	527.51	284.66	0%
ICM	622.89	5.98	0%	1226.44	6.06	0%

Table 6. Results for the irregular 4th order segmentation models with binary variables. The Lazy Flipper consistently outperforms LBP and ICM in terms of energy and runtime. Due to the irregularity, the construction of a meaningful set of spanning trees is non-trivial. TRBP is therefore not applied.

	$\mathbb{E}\{J\}$	$\mathbb{E}\{t\}$	best
LBP	1053.16	99.67	0%
LF	870.52	26.12	**100%**
ICM	2360.76	79.23	0%
LBP2	1053.16	48.51	0%
LF2	870.52	25.34	**100%**
ICM2	2360.76	55.48	0%

Results for the real-world segmentation models are shown in Tab. 6. The Lazy Flipper consistently outperforms LBP and ICM in terms of energy and runtime. Due to the irregularity of the model, the construction of a meaningful set of spanning trees is non-trivial. TRBP is therefore not applied.

We run 400 LBP and TRBP iterations on the fully connected models and on $\mathrm{GRID}^2(1600, 2)$, 1000 iterations on $\mathrm{GRID}^4(1600, 2)$, 300 iterations on the irregular segmentation models, and 100 iterations, otherwise, with a message damping of 0.3 for all models. Message passing is terminated when the maximal change in all messages is less than 10^{-6}. For the fully connected models, all spanning trees are considered for TRBP. For the grid graphs, we set the probability that a factor appears in a sub-tree to the reciprocal of its order. The heuristic for A* search is based on a fan-graph rooted in the last node. The Lazy Flipper is run with a maximal subgraph size of 6.

4 Conclusion

This article presents an empirical comparison of inference algorithms on graphical models with higher order factors. The experiments show that search-based algorithms such as A* search and the Lazy Flipper are powerful tools which can outperform message passing algorithms in these settings. While the set of experiments is far from being exhaustive, it demonstrates the flexibility and modularity of the OpenGM library, in particular the exchangeability of data structures and inference algorithms. More inference algorithms as well as specialized data structures can therefore be examined in the future.

Acknowledgement. This work is connected to the Heidelberg Research Training Group (GRK 1653) on Probabilistic Graphical Models (`http://graphmod.iwr.uni-heidelberg.de/`). Authors acknowledge corresponding support by the German Research Foundation (DFG).

References

1. Pearl, J.: Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann, San Francisco (1988)
2. Koller, D., Friedman, N.: Probabilistic Graphical Models. MIT Press, Cambridge (2009)
3. Aji, S., McEliece, R.: The generalized distributive law. IEEE Transactions on Information Theory 46(2), 325–343 (2000)
4. Kschischang, F., Member, S., Frey, B.J., Andrea Loeliger, H.: Factor graphs and the sum-product algorithm. IEEE Transactions on Information Theory 47, 498–519 (2001)
5. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. IEEE TPAMI 23(11), 1222–1239 (2001)
6. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts? In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2352, pp. 65–81. Springer, Heidelberg (2002)
7. Schlesinger, D.: Exact solution of permuted submodular minsum problems. In: Yuille, A.L., Zhu, S.-C., Cremers, D., Wang, Y. (eds.) EMMCVPR 2007. LNCS, vol. 4679, pp. 28–38. Springer, Heidelberg (2007)
8. Komodakis, N., Paragios, N.: Beyond Pairwise Energies: Efficient Optimization for Higher-order MRFs (June 2009)
9. Kohli, P., Kumar, M.P., Torr, P.H.: P3 & beyond: Move making algorithms for solving higher order functions. IEEE Transactions on Pattern Analysis and Machine Intelligence 31(9), 1645–1656 (2009)
10. Rother, C., Kohli, P., Feng, W., Jia, J.: Minimizing sparse higher order energy functions of discrete variables. In: CVPR, pp. 1382–1389 (2009)
11. Kohli, P., Ladický, L., Torr, P.H.: Robust higher order potentials for enforcing label consistency. Int. J. Comput. Vision 82(3), 302–324 (2009)
12. Murphy, K.P., Weiss, Y., Jordan, M.I.: Loopy belief propagation for approximate inference: An empirical study. In: Foo, N.Y. (ed.) AI 1999. LNCS, vol. 1747, pp. 467–475. Springer, Heidelberg (1999)

13. Weiss, Y., Freeman, W.T.: On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. IEEE Transactions on Information Theory 47(2), 736–744 (2001)
14. Wainwright, M.J., Jordan, M.I.: Graphical Models, Exponential Families, and Variational Inference. Now Publishers Inc., Hanover (2008)
15. Kolmogorov, V.: Convergent tree-reweighted message passing for energy minimization. IEEE Trans. Patt. Anal. Mach. Intell. 28(10), 1568–1583 (2006)
16. Bergtholdt, M., Kappes, J., Schmidt, S., Schnörr, C.: A study of parts-based object class detection using complete graphs. Int. J. Comp. Vision 87(1-2), 93–117 (2010)
17. Besag, J.: On the statisical analysis of dirty pictures. Journal of the Royal Statistical Society B 48, 259–302 (1986)
18. Andres, B., Kappes, J.H., Koethe, U., Hamprecht, F.A.: The Lazy Flipper: A minimal exhaustive search algorithm for higher order graphical models (2010) (forthcoming)
19. Minka, T., Winn, J., Guiver, J., Kannan, A.: Infer.NET 2.3, Microsoft Research Cambridge (2009), `http://research.microsoft.com/infernet`
20. Mooij, J.M., et al.: libDAI 0.2.4: A free/open source C++ library for Discrete Approximate Inference (2010), `http://www.libdai.org/`
21. Szeliski, R., Zabih, R., Scharstein, D., Veksler, O., Kolmogorov, V., Agarwala, A., Tappen, M., Rother, C.: A comparative study of energy minimization methods for markov random fields with smoothness-based priors. IEEE Trans. Pattern Anal. Mach. Intell. 30(6), 1068–1080 (2008)
22. Kolmogorov, V., Rother, C.: Comparison of energy minimization algorithms for highly connected graphs. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3952, pp. 1–15. Springer, Heidelberg (2006)
23. Tian, T.P., Sclaroff, S.: Fast Globally Optimal 2D Human Detection with Loopy Graph Models. In: CVPR (2010)
24. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proc. 8th Int'l. Conf. Computer Vision, vol. 2, pp. 416–423 (July 2001)