Convex inference for community discovery in signed networks *

Guillermo Santamaría
Universitat Pompeu Fabra
Barcelona, Spain
santamaria.guille@gmail.com

Vicenç Gómez
Universitat Pompeu Fabra
Barcelona, Spain
vicen.gomez@upf.edu

Abstract

In contrast to traditional social networks, signed ones encode both relations of affinity and disagreement. Community discovery in this kind of networks has been successfully addressed using the Potts model, originated in statistical mechanics to explain the magnetic dipole moments of atomic spins. However, due to the computational complexity of finding an exact solution, it has not been applied to many real-world networks yet. We propose a novel approach to compute an approximated solution to the Potts model applied to the context of community discovering, which is based on a continuous convex relaxation of the original problem using hinge-loss functions. We show empirically the benefits of the proposed method in comparison with loopy belief propagation in terms of the communities discovered. We illustrate the scalability and effectiveness of our approach by applying it to the network of voters of the European Parliament that we have crawled for this study. This large-scale and dense network comprises about 300 votings periods on the actual term involving a total of more than 730 voters. Remarkably, the two major communities are those created by the european-antieuropean antagonism, rather than the classical right-left antagonism.

1 Introduction

Many complex human interactions can be represented as social networks: graphs that encode different kinds of relations between a set of actors [1]. The problem of detecting communities in social networks consists in finding a suitable clustering or grouping of the nodes in the graph such that individuals with similar behaviour are grouped together. This is a widely studied problem [2, 3, 4]. Community discovery has become relevant in many areas, including demographic distribution [5], knowledge identification [6] and many more.

Traditionally, community discovery only considered relations of agreement or affinity between members. Many real social networks, however, also contain negative relations representing opposition or disagreement [7, 8]. For these signed networks, the problem of community discovery becomes much more challenging due to the different nature of its interactions. It requires not only different models to understand these relations, but also scalable methods that work well in very large networks.

One of the models that has been recently proposed for this goal is the Potts model [9], originated in statistical mechanics to explain the magnetic dipole moments of atomic spins. The problem of finding communities in a network can be expressed as a probabilistic inference problem using the Potts model in which social actors correspond to atomic spins and their social affinity correspond to the magnetic forces between spins. However, due to the computational complexity of finding an exact solution for this model, it has not been applied to many real networks yet.

*NIPS 2015 Workshop: Networks in the Social and Information Sciences

Table 1: Inference methods for estimating the MAP state of Equation (2)

TYPE	METHODS
Exact	Junction Tree
Variational	MaxProd, Linearized Max Prod, Fast Unfolding, Eigenvector
Stochastic	Monte Carlo Sampling

In this paper, we propose a novel approach to compute an approximate solution to the Potts model applied to the context of community discovering in social networks. The proposed method is based on a continuous convex relaxation of the original Potts model using hinge-loss functions. This relaxation makes the inference problem scalable to large-scale networks while retaining a good approximation accuracy.

We evaluate the performance of our approach on a real-world dataset that we have crawled for this study: the network of voters of the European Parliament. This large dataset includes about 300 votings of the actual term and represents a very interesting voting network: it has more than 730 members, which results in a dataset composed of more than $2 \cdot 10^5$ votes in total and thus around $1.5 \cdot 10^7$ indirect pairwise interactions.

2 Problem Statement

We consider a signed network as a connected undirected graph $G = (V, E)$ whose edges E are labeled by $\{1, -1\}$, representing agreement or disagreement between pair of nodes, respectively. Given $k \in I\!N$, our purpose is to assign each node $v_i, i = 1, \ldots, |V|$ to one of k communities $c \in \{1, \ldots, k\}$, taking into account the signs of the edges. This is called a *k-community configuration*.

Intuitively, given a proper *k-community configuration*, vertices that belong to the same community should have agreement relations while they should disagree with vertices belonging to other communities. We say that a network is k-balanced if and only if there exists a k-community configuration such that every edge between communities is negative and every edge within communities is positive. General networks are *unbalanced*, which means that, given a *k-community configuration*, there are some negative edges *within* clusters and some positive edges *between* clusters. These edges are called *frustrated* edges [10]. The alternative problem is then to find a k-community configuration that minimizes the number of frustrated edges. This can be done by minimizing the function:

$$\sum_{(v_i,v_j)\in E} A^-_{ij}\delta(\sigma_i,\sigma_j) + A^+_{ij}(1-\delta(\sigma_i,\sigma_j)), \tag{1}$$

where $\sigma_i, \sigma_j \in \{1, \ldots, k\}$ are the target variables representing the communities, $\delta(\sigma_i, \sigma_j) = 1$ if $\sigma_i = \sigma_j$ and 0 otherwise and A_{ij} is the i, j-th entry of the adjacency matrix. The first term in the sum counts the number of negative edges within communities and the second one the number of positive edges between communities.

Finding a k-community configuration that minimizes the frustration is equivalent to finding the *maximum a posteriori* (MAP) state of the following probability distribution (also known as Boltzmann distribution) defined over all possible configurations:

$$p(\sigma) = \frac{1}{Z} e^{-\sum\limits_{(v_i,v_j)} A^-_{ij}\delta(\sigma_i,\sigma_j) + A^+_{ij}(1-\delta(\sigma_i,\sigma_j))} \tag{2}$$

This is known in the literature as the *Potts Model* [11, 9].

The main problem is that computing the MAP state of Equation (2) is an inference task which is computationally hard, since it requires in the worst case to evaluate an exponential number of assignments. It can not be done in general using exact methods. An exact method and some approximate inference methods [12] are outlined in table 1. While all of them can be used to compute or approximate Equation (2), they are general purpose inference methods and are not intended for the specific task we are considering.

3 A Convex Inference Approach

In this section we propose a novel approach to approximate the MAP state that is based in a relaxation of the original problem as a continuous convex problem. The relaxation expresses the problem as a combination of hinge-loss functions. The hope is that the solution to this new continuous problem is a good approximation to the hard, discrete problem.

3.1 Probabilistic similarity logic

Hinge-Loss functions are at the core of *Hinge-Loss Markov Random Fields* (HL-MRFs), a recent method introduced in [13] to make probabilistic inference tractable on relational networks. It has been successfully used in collective classification [14], knowledge graph identification [15] or sentiment analysis in social networks [16].

HL-MRFs were motivated in *Probabilistic Similarity Logic* (PSL) [17], a general-purpose framework for probabilistic reasoning about similarity in relational domains such as the social networks we are considering. PSL represents the entities of the domain as logical atoms and allows to define first order logical rules that capture the dependency structure of these entities. Some of these rules are *soft* in the sense that they may not be completely fulfilled, but we just look for the joint state that better satisfies them. Other rules are *hard* in the sense that they must hold for a state being considered. Furthermore, each soft rule has a non-negative weight that captures the rule relative importance. Based on this structure, PSL builds a joint probabilistic model over all atoms. The inference problem is to find the most probable state of the *open* atoms given the evidence.

The problem of finding a k-community configuration that minimizes the frustrated edges can be naturally expressed in the PSL language, defining one atom for each possible community each node can belong to: $belongTo(v_i, c)$ for a node v_i of G and a community $c \in \{1, \dots, k\}$. Furthermore, if two nodes v_i and v_j are connected by a positive edge, we will use the expression $positive(v_i, v_j)$, whereas if they are connected by a negative edge, we will use $negative(v_i, v_j)$. Observe that, for both expressions, the symmetric case has to be considered, since the network is undirected. We thus define the following rules:

- If v_i and v_j are connected by a positive edge, the community of v_i and v_j should match, so we define the *soft* rule: $belongTo(v_i, c) \wedge positive(v_i, v_j) \wedge i \neq j \Rightarrow belongTo(v_j, c)$, for each community c.
- If v_i and v_j are connected by a negative edge, the community of v_i and v_j should be different, so we define the *soft* rule: $belongTo(v_i, c) \wedge negative(v_i, v_j) \wedge i \neq j \Rightarrow \neg belongTo(v_j, c)$, for each community c.
- Since every community should have at least one node, we want to define a prior by assigning an initial node v_i^c to each community c, thus we define the *soft* rules $belongTo(v_i^c, c) \Leftrightarrow TRUE$ and $belongTo(v_i^c, c') \Leftrightarrow FALSE$, for $c \neq c'$, for each community c.
- Finally, since each node belongs to exactly one community, we state the *hard* rule: $belongTo(v_i, c) \Leftrightarrow \bigwedge_{c \neq c'} \neg belongTo(v_i, c')$ that basically ensures that the *soft truth values* of $belongTo(v_i, c)$ for all possible c will sum up 1.

3.2 Hinge-Loss Markov Random Fields

We have defined our relational domain by specifying the soft and hard rules over the set of atoms. The atoms can be seen as a continuous relaxation of the original discrete binary variables, which now take values in the range $[0, 1]$. We now define a joint probability distribution over these different *states* of our problem. In particular, we want to define a probability function whose MAP state is the one which satisfies *best* the *soft* rules (we will talk about the hard ones in a moment). In other words, this probability function will measure the *distance to satisfaction* of the soft rules. Note that these rules behave as boolean logic with the difference that their atoms can take continuous values instead of binary ones. Following this reasoning, it means that the rule's implication would be satisfied when the value of the antecedent is equal or larger than the value of the consequent. This can be expressed using the following functions:

Definition 1. *Let $x = (x_1, \dots, x_n)$ be a vector of $\mathbb{R}^n$. A hinge-loss function is a function of the form $\Phi(x) = \max\{l(x), 0\}^m$, where l is a linear function and $m \in \mathbb{N}^*$.*

Although hinge-loss functions are not strict metric functions, they are *convex* and a linear combination of hinge-loss function can actually become a metric. For these reasons, they were suggested in [13] to measure the distance to satisfaction of the soft rules. In particular, for a soft rule R, we define the hinge-loss function:

$$d(R) : \max\left\{\bigwedge(x_{B_1},\dots,x_{B_r}) - \bigvee(x_{H_1},\dots,x_{H_s}), 0\right\}^m \tag{3}$$

where x_{B_i}, x_{H_j} are the variables of atoms B_i, H_j in the space of states and $\bigwedge(x_{B_1},\dots,x_{B_r})$ and $\bigvee(x_{H_1},\dots,x_{H_s})$ are functions that make $d(R)$ hinge-loss. So far, we have only considered soft rules. As stated before, *hard* rules are those that need to be fully satisfied by the solution. They can be considered as constraints on the domain of states that are translated to linear equalities/inequalities using $\wedge$ and $\vee$.

In conclusion, inference on a PSL program (also known as *hinge-loss minimization*) is equivalent to finding the state that minimizes the *distance to satisfaction* function induced by the *soft* rules, in a constrained domain of $[0,1]^n$, defined by linear equalities and inequalities induced by the *hard* rules. The following definition summarizes this:

Definition 2. *Let $x = (x_1,\dots,x_n)$ be a vector of $\mathbb{R}^n$. Let $\{\Phi_i(x)\}_{i=1}^r$ be a a set of hinge-loss functions over x and $\{\lambda_i\}_{i=1}^r$ positive weights corresponding to each rule r. Let $\mathcal{X}$ be a domain defined by $\mathcal{X} = \{x \in \mathbb{R}^n | g_i(x) \le 0, f_j(x) = 0\}$ where $g_i(x)$ and $f_j(x)$ are linear functions. The hinge-loss minimization problem is defined by:*

$$\min \sum_{i=1}^{r} \lambda_i \Phi_i(x), \qquad \textit{subject to } x \in \mathcal{X}$$

where λ_i is the positive weight of the i-th hinge-loss term.

3.2.1 Expressing the Potts Model as a Hinge-Loss Problem

Now that we have formulated the *hinge-loss minimization problem* and showed its equivalence to an inference problem, we translate the PSL rules defined in the previous section to a hinge-loss minimization problem.

Given a number of communities k and a k-community configuration, for a node v_i, let σ_i be a random variable defined by the community of v_i. Let $x_i^c = p(\sigma_i = c)$. Then:

- For each positive edge (v_i, v_j), we define the hinge-loss functions $\max\{x_i^c - x_j^c, 0\}$ and $\max\{x_j^c - x_i^c, 0\}, \forall c \in \{1,\dots,k\}$
- For each negative edge (v_i, v_j) define the hinge-loss functions $\max\{x_i^c - x_j^d, 0\}$ and $\max\{x_j^c - x_i^d, 0\}, \forall c, d \in \{1,\dots,k\}, c \neq d$
- Additionally, for each community c, assign an initial node v_0 to c, by defining the hinge-loss function $\max\{1 - x_0^c, 0\}$ and $\max\{x_0^d, 0\}, \forall c \neq d$
- We choose all weights λ_i to be equal to 1 for the sake of simplicity.

Putting all together, we have the following hinge-loss minimization problem:

$$\min \sum_{i,j} (A_{ij}^- \sum_{c=1}^{k} \sum_{d=1, d\neq c}^{k} \text{SHL}(x_i^c, x_j^d)^{m_1} + A_{ij}^+ \sum_{c=1}^{k} \text{SHL}(x_i^c, x_j^c)^{m_2}) + \sum_{c=1}^{k} \text{SHL}(x_0^c, 1)^{m_3}$$

$$\text{subject to } \sum_{c=1}^{k} x_i^c = 1, 0 \le x_i^c, i = 1,\dots,|V|$$

where $\text{SHL}(x,y)^m = \max\{x-y,0\}^m + \max\{y-x,0\}^m$.

Given a vertex v_i and a solution to the previous problem, we assign v_i to the most likely community, i.e. $\widetilde{\sigma}_i = \{\widetilde{c} | x_i^{\widetilde{c}} = \max_c x_i^c\}$.

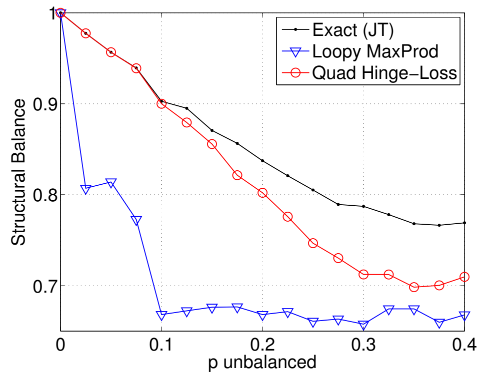

Figure 1: *Results for $k = 2$ as a function of the network unbalance. The hinge-loss method approximates better the structural balance of the synthetic network compared to Loopy MaxProd.*

3.2.2 Finding a Global Optimum

To find the optimum of the previous convex minimization problem, we use the *Alternating Direction Method of Multipliers* [18]. This method breaks the original problem into a set of easier convex optimization problems, the complexity of each one depends on m, the exponent of the hinge terms. We use the variant presented in [19], which contains an improvement for $m \in \{1, 2\}$ that allows to perform the optimize in very large networks.

We will see in the next section that taking $m_1 = m_2 = m_3 = 2$ works much better, probably because it penalizes more the *neutral* state, where $x_i^c = 1/k$.

4 Experiments with Synthetic Networks

Now we evaluate our proposed method in networks generated using realistic models of network growth. They provide a ground truth to quantify the impact of the different parameters of the model and the robustness of the method in different conditions, such as the number of underlying communities, their distribution or the proportion of negative and positive edges. We construct a signed network with given k communities by adapting the method proposed by [20]. We apply our method and compare its results against two other inference algorithms: the junction tree (JT) [21], which provides exact results but it is only feasible in small networks and loopy belief propagation [22], which provides approximate solutions, but can be applied to larger networks. BP provides exact marginals on trees, but can also be useful as an approximate algorithm on general graphs with cycles [12, 23]. Since we are interested in approximating the MAP state instead of the single variable marginals, we use a variant which replaces the sum operator with a max, known as Loopy Max-Product (or Min-Sum) algorithm and we will refer to it as Loopy MaxProd.

We first evaluate our method *Hinge-Loss* on networks with $k = 2$ underlying communities and several configurations of parameter values. We start by analyzing the accuracy of *Hinge-Loss* as we vary the parameter $p_{\text{unbalanced}}$: the amount of unbalance of the network between the underlying communities. For large values of $p_{\text{unbalanced}}$, the proportion of frustrated edges is higher and thus the problem becomes more difficult. We measure performance in terms of the *structural balance*, introduced in [24]: this is just the proportion of *non frustrated edges* for a concrete k-community configuration, i.e. the number of non frustrated edges divided by the total number of edges. We obtain the ground truth using the *Exact* method and another approximation using *Loopy MaxProd*.

We show results for $n = 100, k = 2$ and a quadratic form, which gives a much better performance: in fact is always better than the result produced by *Loopy MaxProd*, as showed in Figure 1. This is probably due to the fact that quadratic hinge-loss terms penalize more those communities configurations that contain negative edges within communities.

It has been established a mean unbalance value of $p_{\text{unbalanced}} \approx 0.22$ in real social networks [24]. Observe that for this value we have that the hinge-loss approximations is a 0.93 of the exact result.

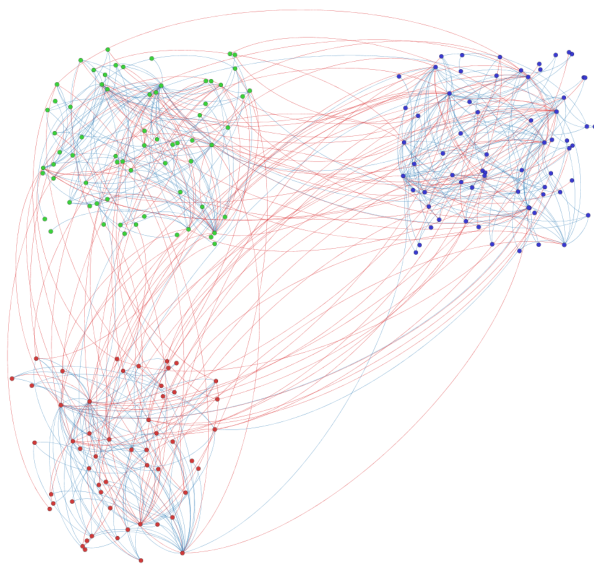

Figure 2: *Example of community structure generated for* $p_{unbalanced} = 0.15$. Positive edges are depicted in blue and negative edges in red. The color of the node indicates the community membership.

Detecting three or more communities is a much more challenging task since the problem is not a binary classification problem any more. However, in the experiments done, we have proved that the performance of our method is always better than *Loopy MaxProd*. Figure 2 shows an example of a generated network for a $k = 3$ communities.

Regarding computational complexity and scalability of the algorithm, solving a hinge-loss minimization problem using the Alternating Direction Method of Multipliers tends to be linear in the number of rules of the problem, and this has been proved by our experiments [19]. In our case, the complexity is of the order of the number of edges times the square of number of communities, $\mathcal{O}(mk^2)$. For a fixed number of communities this complexity is equivalent to the linear approximation of loopy BP proposed in [25]. So, our method is not only better than loopy BP but its complexity is equivalent to the complexity of some of its approximations.

5 The community structure of the European parliament

Finally, we analyze the network induced by the votings on the European Parliament. The goal of this analysis is to discover different underlying communities without using the actual party memberships.

The European Parliament represents a very interesting voting network: it has 751 members and 8 political groups. The voting process does not have *party discipline*, i.e. the members can be unloyal to their political group in some votings. We focus in 730 members only, since 21 of them were substituted in the middle of the term.

We have scrapped 300 votings events from the actual term, from May 2014 to June 2015 [1]. In a voting event, each parliament member has one of the following possibilities: it can vote *for*, vote *against*, *abstain*, *not vote* and *be absent*. For simplicity, we only consider *for* and *against* votes.

We first combine all the votings events to represent the European Parliament as a signed network. We give less relevance to the votings with large agreement between the members by considering the *discrepancy* of a voting, which is defined as the entropy of a voting event.

[1] We crawled the website http://www.votewatch.eu/ to obtain the data.

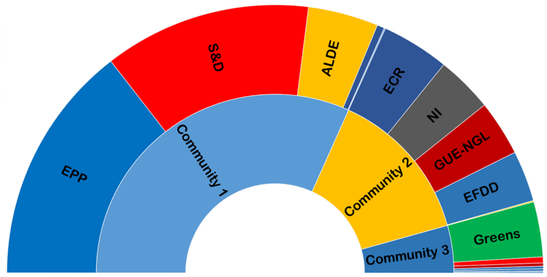

Figure 3: Communities obtained by our method for $k = 3$.

Let n_f^r and n_a^r be the number of *for* and *against* votes in the voting r, respectively ($n_{tot}^r = n_f^r + n_a^r$). We define the *discrepancy* of the voting r as

$$H_r = -\frac{n_f^r}{n_{tot}^r} \log \frac{n_f^r}{n_{tot}^r} - \frac{n_a^r}{n_{tot}^r} \log \frac{n_a^r}{n_{tot}^r}.$$

For two different parliament members v_i, v_j, we define:

$$\lambda(i,j)^r = \begin{cases} H_r \frac{n_a^r}{n_{tot}^r}, & \text{if both voted for} \\ H_r \frac{n_f^r}{n_{tot}^r}, & \text{if both voted against} \\ -H_r \max\left(\frac{n_f^r}{n_{tot}^r}, \frac{n_a^r}{n_{tot}^r}\right), & \text{if voted different} \\ 0, & \text{otherwise.} \end{cases}$$

We compute the edge sign between members v_i and v_j by thresholding the sum of $\lambda(i,j)^r$ over all votings r. After this, the resulting network has $92,810$ positive edges and $111,579$ negative edges.

The results are very enlightening and even surprising from the social point of view, specially when compared with the actual political groups. We summarize them:

For $k = 2$, we get very differentiated blocks: those created by the european-antieuropean antagonism, rather than the classical right-left antagonism. The case of the *Green party* is very interesting: although it is theoretically in the opposition, in our community configuration is included in the block of the commission majority, with the socialists, ALDE, and the popular party.

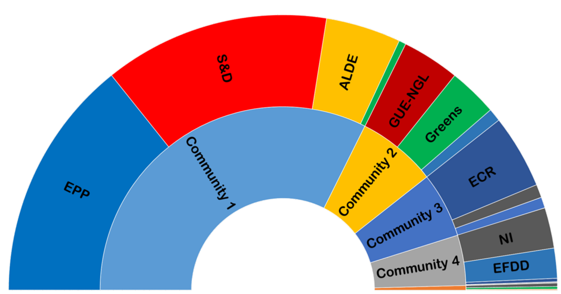

Figure 4: Communities obtained by our method for $k = 5$.

For $k = 3$, the *pro-european* block breaks down as shown in Figure 3: the Green party *feels uncomfortable* with this right and center political parties, but it is not as radical as the parties in the other group, so almost the entire party forms a separated community while the anti-european block remains stable.

For $k = 4$, one of the communities is conformed by the socialists, ALDE and the popular party, but now the anti-european block starts to break. In particular, the GUE-NGL, a radical left party, is in one separated community. This makes sense, since the rest of the parties of the anti-european block are conservatives (ECR) or even more right wing (NI).

For $k = 5$ (Figure 4), the pro-european group remains stable, but now the green and the GUE-NGL parties (both left-wing) belong to the same community. On the other hand, the ECR party forms one separated community, and the NI and EFDD parties (both right-wing) remain in the same community. Finally we have a very small community that contains a mixture of different parties. So our algorithm is able to detect the european-antieuropean antagonism but also the right-left antagonism.

We can conclude that the proposed approach is an efficient method to discover groups that are hidden and become visible at different scales.

6 Conclusions

Discovering community structure in signed networks is one of the most important open problems in order to better understand network function and evolution mechanisms. We have presented a method to compute an approximate solution to the Potts model based on a continuous convex relaxation of the original problem using hinge-loss functions. We have shown the effectiveness of the proposed method in artificial and real networks, including the large European parliament network. These results help to understand the behaviour of the political european groups. Our method is able to discover non-trivial communities in terms of the classical antagonism, as well as by their european-antieuroepean antagonism. The proposed methodology allows to discover this structure efficiently and this makes our approach potentially applicable in other large networks.

References

[1] J. Bruggeman. *Social networks: An introduction*. Routledge, 2013.

[2] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3):75–174, 2010.

[3] A. Lancichinetti and S. Fortunato. Community detection algorithms: a comparative analysis. *Physical review E*, 80(5):056117, 2009.

[4] G. Palla, I. Derényi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005.

[5] A. L. Traud, P. J. Mucha, and M. A. Porter. Social structure of facebook networks. *Physica A: Statistical Mechanics and its Applications*, 391(16):4165–4180, 2012.

[6] M. EJ Newman. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences*, 98(2):404–409, 2001.

[7] M. EJ. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[8] A. Clauset, M. EJ. Newman, and C. Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

[9] R. B. Potts. Some generalized order-disorder transformations. *Proc. the Cambridge Philosophical Society*, 48:106–109, 1952.

[10] V. A. Traag and J. Bruggeman. Community detection in networks with positive and negative links. *Physical Review E*, 80(3):036115, 2009.

[11] F.-Y. Wu. The potts model. *Reviews of modern physics*, 54(1):235, 1982.

[12] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2):1–305, 2008.

[13] S. Bach, B. Huang, B. London, and L. Getoor. Hinge-loss Markov random fields: Convex inference for structured prediction. *arXiv preprint arXiv:1309.6813*, 2013.

[14] B. London, S. Khamis, S. H. Bach, B. Huang, L. Getoor, and L. Davis. Collective activity detection using hinge-loss Markov random fields. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2013 IEEE Conference on*, pages 566–571. IEEE, 2013.

[15] J. Pujara, H. Miao, L. Getoor, and W. Cohen. Knowledge graph identification. In *The Semantic Web–ISWC 2013*, pages 542–557. Springer, 2013.

[16] R. West, H. S. Paskov, J. Leskovec, and C. Potts. Exploiting social network structure for person-to-person sentiment analysis. *arXiv preprint arXiv:1409.2450*, 2014.

[17] M. Brocheler, L. Mihalkova, and L. Getoor. Probabilistic similarity logic. *arXiv preprint arXiv:1203.3469*, 2012.

[18] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[19] S. Bach, M. Broecheler, L. Getoor, and D. O'leary. Scaling MPE inference for constrained continuous Markov random fields with consensus optimization. In *Advances in Neural Information Processing Systems*, pages 2654–2662, 2012.

[20] P. L. Krapivsky and S. Redner. A statistical physics perspective on web growth. *Computer Networks*, 39(3):261–276, 2002.

[21] S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 157–224, 1988.

[22] A. T. Ihler, J. Iii, and A. S. Willsky. Loopy belief propagation: Convergence and effects of message errors. In *Journal of Machine Learning Research*, pages 905–936, 2005.

[23] V. Gómez, H. J. Kappen, and M. Chertkov. Approximate inference on planar graphs using loop calculus and belief propagation. *J. Mach. Learn. Res.*, 11:1273–1296, August 2010.

[24] G. Facchetti, G. Iacono, and C. Altafini. Computing global structural balance in large-scale signed social networks. *Proceedings of the National Academy of Sciences*, 108(52):20953–20958, 2011.

[25] W. Gatterbauer, S. Günnemann, D. Koutra, and C. Faloutsos. Linearized and single-pass belief propagation. *Proceedings of the VLDB Endowment*, 8(5):581–592, 2015.