2013 IEEE INTERNATIONAL WORKSHOP ON MACHINE LEARNING FOR SIGNAL PROCESSING, SEPT. 22–25, 2013, SOUTHAMPTON, UK

# LARGE SCALE INFERENCE IN THE INFINITE RELATIONAL MODEL: GIBBS SAMPLING IS NOT ENOUGH.

*Kristoffer Jon Albers, Andreas Leon Aagard Moth, Morten Mørup, Mikkel N. Schmidt*

Section for Cognitive Systems
DTU Compute
Technical University of Denmark

## ABSTRACT

The stochastic block-model and its non-parametric extension, the Infinite Relational Model (IRM), have become key tools for discovering group-structure in complex networks. Identifying these groups is a combinatorial inference problem which is usually solved by Gibbs sampling. However, whether Gibbs sampling suffices and can be scaled to the modeling of large scale real world complex networks has not been examined sufficiently. In this paper we evaluate the performance and mixing ability of Gibbs sampling in the Infinite Relational Model (IRM) by implementing a high performance Gibbs sampler. We find that Gibbs sampling can be computationally scaled to handle millions of nodes and billions of links. Investigating the behavior of the Gibbs sampler for different sizes of networks we find that the mixing ability decreases drastically with the network size, clearly indicating a need for better sampling strategies.

***Index Terms***— Infinite Relational Model, Markov Chain Monte Carlo, Gibbs sampling, large scale network modelling, Bayesian inference.

## 1. INTRODUCTION

In our every day lives, we constantly interact with systems that can be described and examined as complex networks, including both natural and human systems. Among the human created systems, we encounter supply chains, power grids and internet communication, while in the natural world many biomechanical and behavioral systems can be described as complex networks, such as neural interaction in the brain and social relations between people. Hence complex networks are a very integrated part of the world and the way humans act and adapt. Being able to comprehend and develop theoretical models to describe these complex networks is an emerging and intriguing science, where we need to combine experiences and knowledge from many different research areas [1].

Some approaches to model complex networks use Bayesian statistical modelling in order to account for interactions in the network. One of these approaches are the stochastic block-models [2] including the Infinite Relational Model (IRM) [3, 4]. In this paper we use IRM combined with Gibbs sampling, which is often the workhorse of Bayesian inference, see also [5, 6, 7, 8, 9]. We have chosen IRM as it has become a very prominent network model.

Statistical analysis of large scale networks is an active field of research. In [6] large scale MCMC was developed for the stochastic relational model to allow efficient computation on a bipartite network with $\sim$500 thousand nodes and $\sim$100 million links. In [7] GPU resources were utilized to perform bipartite co-clustering by the IRM model. These studies were conducted on graphs in the order of $\sim$8 million nodes and $\sim$500 million links. Although, these works have given valuable insight to the robustness and scalability of the models, they leave room for further investigation into larger and more complex networks. Furthermore, inference in bipartite networks is inherently easier to parallelize than the modeling of unipartite networks where the interaction between entities no longer decouples.

In this paper we go an order of magnitude beyond current modeling limitations for the size of the networks and we consider the more challenging case of unipartite networks. We have developed and implemented a program able to perform fast IRM analysis on unipartite networks with millions of nodes and billions of links. We test how IRM scales by running our program on different sized graphs based on the modeling of a single large social network with $\sim$65.5 million nodes and $\sim$1.8 billion links including subsamples thereof. The network represents user relations on the social network Friendster and is maintained by the Stanford Network Analysis Platform (SNAP) (http://snap.stanford.edu/data/com-Friendster.html). We further investigate how IRM performs when run for millions of Gibbs sweeps on a relatively small network of structural brain connectivity [10] containing $\sim$1000 nodes.

The contribution of this paper is to demonstrate that the Gibbs sampler for the Infinite Relational Model can be scaled up to the modelling of millions of nodes and billions of links. Being able to sample with high computational performance

This work was supported by the Lundbeck Foundation.

978-1-4673-1026-0/12/$31.00 

we set out to test the limitations of the Gibbs sampler for the IRM running millions of Gibbs iterations on networks of variable sizes. In particular, we determine the network size beyond which the Gibbs sampler fails, i.e. is unable to properly mix over the posterior distribution. Surprisingly, this seem to already be an issue for networks of $\sim$1000 nodes.

## 2. METHOD

### 2.1. Infinite Relational Model

The purpose of the stochastic blockmodel is to split the vertices in a graph into smaller subsets called blocks, such that these blocks capture the overall clustering structure of the graph [2]. The infinite relational model as proposed in [3] and [4] is an extension to the stochastic blockmodel, by allowing an unlimited number of clusters, determined by a Chinese Restaurant Process (CRP). As a generative model, IRM relies on the following generative process:

$$\boldsymbol{z} \sim \mathrm{CRP}(\alpha), \qquad \text{groups} \tag{1}$$

$$\eta_{lm} \sim \mathrm{Beta}(\beta^+, \beta^-), \qquad \text{interactions} \tag{2}$$

$$A_{ij} \sim \mathrm{Bernoulli}(\eta_{z_i z_j}), \qquad \text{links.} \tag{3}$$

The clustering follows the Chinese Restaurant Process. The probability of connection between nodes in two clusters $l$ and $m$ follows the beta distribution, while the links between individual nodes are based on the Bernoulli distribution. The hyperparameters $\alpha, \beta^+$ and $\beta^-$ are in the following represented by $h$. By collapsing $\boldsymbol{\eta}$ we obtain the following joint likelihood according to the generative process:

$$\begin{aligned} P(\boldsymbol{A}, \boldsymbol{z}|h) &= \int P(\boldsymbol{A}, \boldsymbol{z}, \boldsymbol{\eta}|h) d\boldsymbol{\eta} \\ &= \frac{\alpha^K \Gamma(\alpha) \prod_k \Gamma(n_k)}{\Gamma(J-\alpha)} \prod_{l \leq m} \frac{\mathrm{B}(N^+_{lm} + \beta^+, N^-_{lm} + \beta^-)}{\mathrm{B}(\beta^+, \beta^-)} \end{aligned} \tag{4}$$

where $K$ is the number of clusters, $J$ is the total number of nodes, $n_k$ is the number of nodes in cluster $k$, while $N^+_{lm}$ and $N^-_{lm}$ are the number of links and non-links between cluster $l$ and $m$ respectively. $\mathrm{B}(a,b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$ is the beta function. Inferring this posterior distribution is a computationally hard problem. However, the problem can be addressed estimating the cluster assignment using Markov Chain Monte Carlo (MCMC) inference by Gibbs sampling, as proposed as Bayesian estimator for Stochastic Blockmodels in [2] and described for IRM in [5, 3].

### 2.2. Markov Chain Monte Carlo

The idea behind MCMC is to iteratively change the model in a way such that in the limit of infinitely many changes correct draws from the full posterior distribution are generated.

```
1 for each iteration
2     for each node i
3       calculate N+ and N-, n
            ignoring i
4       calculate r for the node
5       calculate the probability
            change of the model when
            assigning i to each cluster
6       choose a cluster assignment
            based on these probabilities
```

**Fig. 1**. Naive IRM Pseudo-code.

A special case of MCMC is Gibbs sampling where each parameter is changed one at a time by generating new values from its posterior marginal distribution. For the IRM this corresponds to calculating the posterior marginal distribution of assigning each node to each of the existing groups or to a new empty group. This procedure is repeated, sweeping over all parameters [5]. Assigning a node $i$ to a cluster $o$, changes the likelihood in equation (4) such that the first fraction (given by the CRP), is multiplied by $\alpha$ if $o$ is a new empty cluster and by the number of nodes in cluster $o$ ignoring node $i$ ($n_{o\setminus i}$) otherwise. Furthermore, the product in equation (4) will be multiplied with

$$\prod_m \frac{B(N^{+\setminus i}_{om} + r_{im} + \beta^+, N^{-\setminus i}_{om} + n_m - r_{im} + \beta^-)}{B(N^{+\setminus i}_{om} + \beta^+, N^{-\setminus i}_{om} + \beta^-)}, \tag{5}$$

where $r_{im}$ is the total sum of links between node $i$ and all nodes in cluster $m$. Using Bayes theorem we can calculate the probability of assigning a single node $i$ to a cluster $o$ as:

$$P(z_i = l|\mathbf{A}, \mathbf{z}_{\setminus i}, h) = \frac{P(\mathbf{A}, \mathbf{z}_{\setminus i}, z_i = l|h)}{\sum_{o=1}^{K+1} P(\mathbf{A}, \mathbf{z}_{\setminus i}, z_i = o|h)}. \tag{6}$$

In order to ensure numeric stability at the level of machine precision this posterior is calculated in the log domain. Thus, the key operation necessary for the evaluation of this posterior is the calculations of the logarithm of the beta function.

## 3. LARGE SCALE IMPLEMENTATION

To implement the Gibbs sampler efficiently such that it can feasibly be run on large scale problems it is essential that the algorithm is implemented using an efficient data structure avoiding expensive memory access. In addition it is important to implement the logarithm of the beta function in a cost efficient manner.

Our starting point is the naive pseudo-code implementation of IRM using MCMC with Gibbs sampling given in Figure 1. The inner loop performs a Gibbs-sweep iteratively as-

```
1 Initialize N+, N− and n
2 for each iteration
3     for each node i
4       calculate r for the node
5       calculate the probability
            change of the model when
            assigning i to each cluster
6       choose a cluster based on these
            probabilities
7       if i has changed cluster, update
            N+, N− and n
```

**Fig. 2**. Optimized IRM Pseudo-code.

signing each node to a cluster. This is part of the Gibbs sampler while the outer loop performs each Gibbs sweep continuously sampling the model as part of the MCMC procedure. To perform each Gibbs update the number of links, non-links, nodes in each cluster, and connections $r$ are computed after which the probability of each group assignment is computed and normalized following the conditional posterior given in equation (6). From these probabilities a new cluster assignment is chosen for the current node, and the procedure is continued for the next node.

In each iteration of the Gibbs-sweep, exactly $2 \times K$ values in $N^+$ and $N^-$ are changed when a node is reassigned. Instead of recomputing these values, they can be cached globally and simply updated in each iteration. The same holds true for $n$, where only two values change. All $r$-vectors could also be cached and updated between each iterations. However the collective size of all $r$-vectors will be $J \times K$, which means the computer would need 24 GB for 65 million nodes and only 100 clusters, which might not be feasible. On the other hand, $N^+$, $N^-$ and $n$ can easily be cached.

To avoid computing unnecessary changes to the data structures, we only update $N^+$ and $N^-$ after finding the cluster the node is assigned to. If the node does not change cluster, no updates are necessary. This affects the probability calculation, as the node is not removed from $N^+$, $N^-$ and $n$ before the probabilities are calculated. However, this only affects the calculation for the assigned cluster and is very easy to take into account. The product over cluster $c$ in formula (5) changes to

$$\prod_m \frac{B(N_{oc}^+ + \beta^+, N_{oc}^- + \beta^-)}{B(N_{oc}^+ - r_{ic} + \beta^+, N_{oc}^- - (n_c - r_{ic}) + \beta^-)}, \qquad (7)$$

while one is subtracted from $n_c$ whenever it is accessed. This is mathematically equivalent to the original formulation but working on this parametrization will result in a more efficient memory access pattern. Furthermore, each term of the sum can be computed in parallel. The calculation of $r$ as well as the updates of $N^+$ and $N^-$ can also be parallelized. We implement these parallelizations using the OpenMP API, resulting in a more efficient ressource utilization. The pseudo-code for using these optimizations is shown in figure 2.

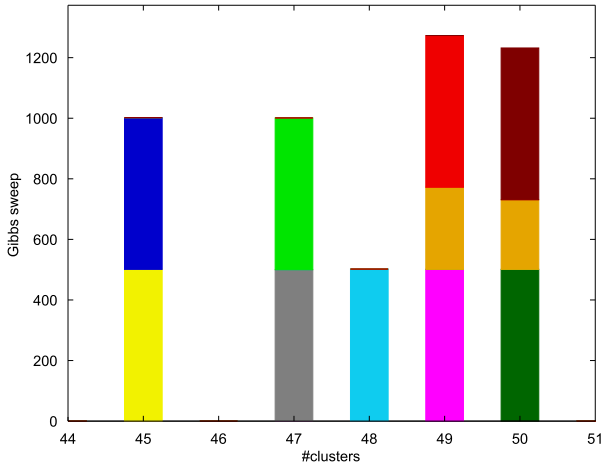


**Fig. 3**. Times a number of clusters were found in last 5M Gibbs sweeps (sub-sampled every 1000th sweep). Color index indicates different runs.

Aside from optimizing the structure of the program, there are also many other optimizations, which can greatly improve the speed of the program. One of the most prominent speed reductions come from cache misses, which occurs when data is not accessed sequentially, but randomly in the memory. To avoid cache misses, we store the entire matrix for $N^+$ and $N^-$ even though both matrices are symmetric. If only the upper triangle of $N^+$ and $N^-$ were saved, all data associated with a cluster could not be stored in sequence.

Another improvement is pre-calculations. The program computes the logarithm of the beta function, which is rather slow. For a network with 65 million nodes, the range of the input parameters are in the interval 1 to $65.000.000^2 \sim 4.2 \times 10^{15}$. We have optimized the computations of the logartihm of the gamma function used in the computation of the log-beta function by precalculating and caching the function for the first ∼250M values and use Stirling's approximation for large values where it is correct with high precision.

## 4. RESULTS AND DISCUSSION

In order to evaluate how IRM performs, we have evaluated both a medium sized network and a group of subsampled networks. These networks represents actual networks and hence provides more realistic results, compared to synthetic data.

### 4.1. Large number of Gibbs sweeps

To evaluate how IRM performs for millions of Gibbs sweeps, we considered the five graphs of structural brain connectivity across 998 brain regions (nodes) [10] derived by tractography

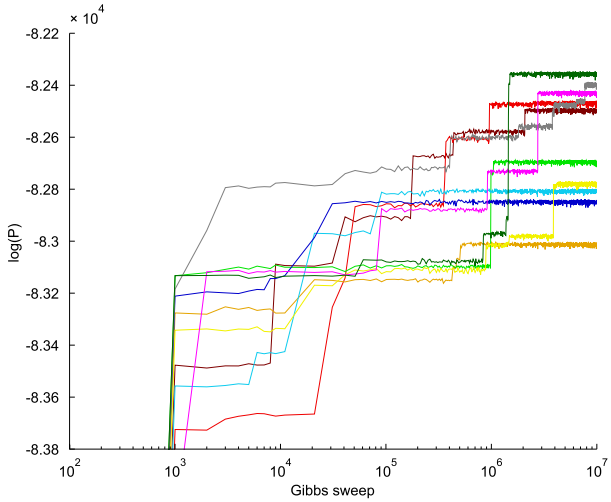


**Fig. 4**. Log-likelihood for different Gibbs sweeps. Color index indicates different runs.

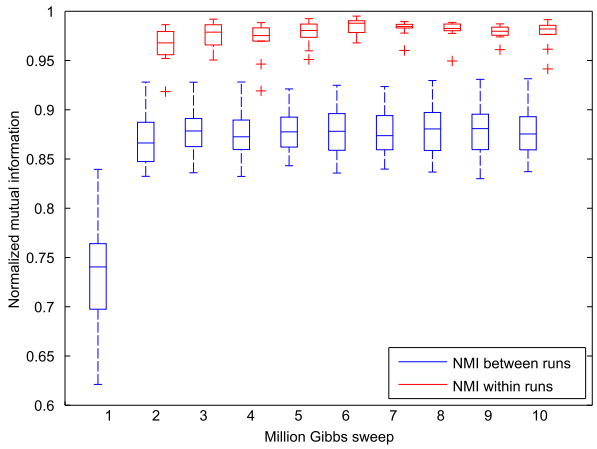


**Fig. 5**. Normalized Mutual Information between and within the 10 runs.

on diffusion spectrum imaging (we considered the non-zero elements of the first scans averaged and symmetrized over the five subjects). 10 different runs were performed on this graph for 10M Gibbs sweeps each, with the hyperparameters $\beta^+ = 1$, $\beta^- = 1$ and $\alpha = 6$.

The number of clusters found by the different runs is shown in figure 3. This figure shows how often each run contained a particular number of clusters during the last 5M Gibbs sweeps. The runs do not converge to the same distribution of number of components, but ranges from 45 to 50 components. Furthermore, all runs except one appears to have settled on a single number of clusters. This indicates a poor mixing ability of the Gibbs sampler.

To evaluate, whether the sampler in fact performs poor mixing, we have plotted the log-likelihood as a function of the number of Gibbs sweeps, using equation (4), shown in figure 4. From this plot we see that the runs do in fact not mix, even after 10 million sweeps. To further investigate this, we have looked at the normalized mutual information (NMI) within and between runs, shown in figure 5. We used $\mathrm{NMI}(\mathbf{z}', \mathbf{z}'') = \frac{2\mathrm{I}(\mathbf{z}', \mathbf{z}'')}{\mathrm{H}(\mathbf{z}')+\mathrm{H}(\mathbf{z}'')}$, where $\mathrm{I}(\mathbf{z}', \mathbf{z}'')$ is the mutual information between two groupings of the nodes $\mathbf{z}'$ and $\mathbf{z}''$ and $\mathrm{H}(\mathbf{z}')$ is the entropy of $\mathbf{z}'$. To evaluate the normalized mutual information within runs, each run has been compared to its configuration 1 million sweeps earlier. The first NMI within runs is not visible on this plot, as it is extremely close to 0 as each of the runs are more similar with themselves than each other. Additionally the similarity between runs remains consistent over time, indicating that the runs will not reach consensus within reasonable time.

### 4.2. Large networks

In order to investigate how the IRM model with Gibbs sampling scales with the size of the network, we have grown several graphs of increasing size, subsampled from the large 65M node Friendster social network dataset available from SNAP. These subsampled graphs contains $10^1, 10^2, 10^3, 10^4, 10^5, 10^6$ and $10^7$ nodes. Each subgraph is grown by initially including the first node. Afterwards the nodes connected to the first included node is added, then the nodes connected to the second included node is added etc. until the specified number of nodes has been selected. This procedure ensures that no node is selected, which is not connected to any other node. Furthermore all minor networks are subsets of the larger networks. Each network has been run 5 times, with the hyperparameters $\beta^+$ and $\beta^-$ set to 1, while $\alpha$ is set to $\lfloor log(n) \rfloor$, where $n$ is the number of nodes in the network.

To compare the runtime of the algorithm for the different sized graphs, one run for each graph were computed on the same computer for 5 hours to at most 1M Gibbs sweeps. All other runs were run the exact same number of Gibbs sweeps.

The runtime of the algorithm as the network size increases is shown in figure 6. The figure also shows the number of Gibbs sweeps the test computer were able to perform within the time limit. From this we see that the number of seconds per Gibbs sweep is linearly bounded by the number of nodes.

Figure 7 shows the log-likelihood for the five runs on different sized networks. Comparing these runs we can see that the Gibbs sampler mixes fairly well for less than 1000 nodes after about 10k Gibbs sweep. However, for larger networks the mixing ability of the Gibbs sampler decreases rapidly.

To further investigate this, we have looked at the normalized mutual information between and within runs for different sized networks, shown in figure 8. We see that the Gibbs sampler mixes well for 10 and 100 nodes, however, the mixing

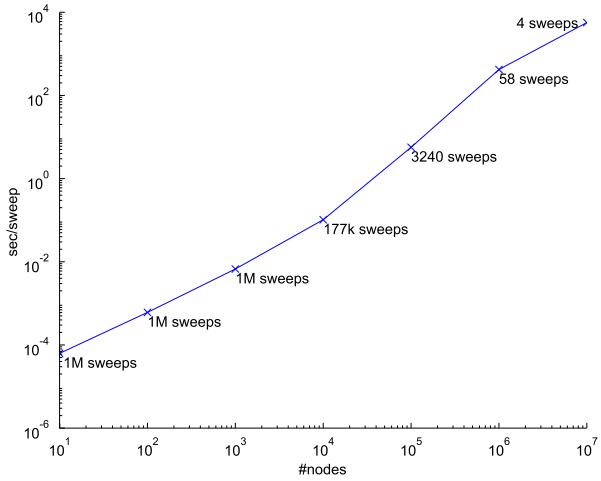


**Fig. 6**. Average time to perform Gibbs sweeps on the different sized networks. For each run is given the number of Gibbs sweeps performed.

ability decreases rapidly for larger networks as indicated by runs being more similar to themselves than others. This corroborates the results of the log-likelihood. Though IRM with Gibbs sampling is able to mix very well for small networks, it is not a sufficient inference procedure for larger networks.

## 5. CONCLUSION

In this article we have demonstrated how IRM with Gibbs sampling behaves on large scale real world complex networks. To do this, we modelled and implemented an algorithm, that allowed us to run the IRM on larger and more complex networks, than to the best of our knowledge has previously been done.

We found that Gibbs sampling may not be sufficient for networks of about 1000 nodes. Even after 10 million Gibbs sweep for 10 distinct runs on the same network, the log-likelihood did not mix for any of the runs. As the number of nodes increased, this effect became even worse. By using Gibbs sampling on grown networks of increasing size, we found that the mixing ability of the Gibbs sampler decreases drastically as the network size increases such that for about 1.000 nodes, the Gibbs sampler failed to properly mix over the posterior distribution.

These findings clearly indicates a need to use other sampling strategies or extend the Gibbs sampler with more complex heuristics or procedures, including split-merge [11], which has previously been utilized for IRM [12, 13, 8].

Care should be taken when using IRM with pure Gibbs sampling on large networks as issues with convergence are prominent.

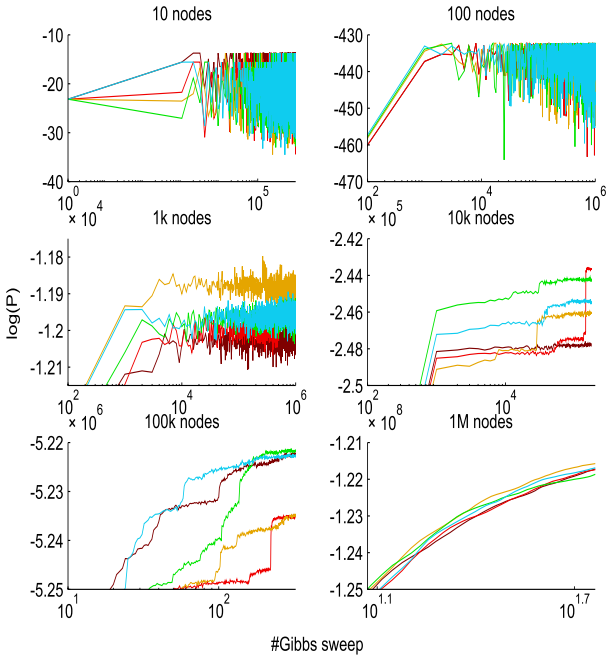


**Fig. 7**. Log-likelihood for multiple runs on different sized networks.

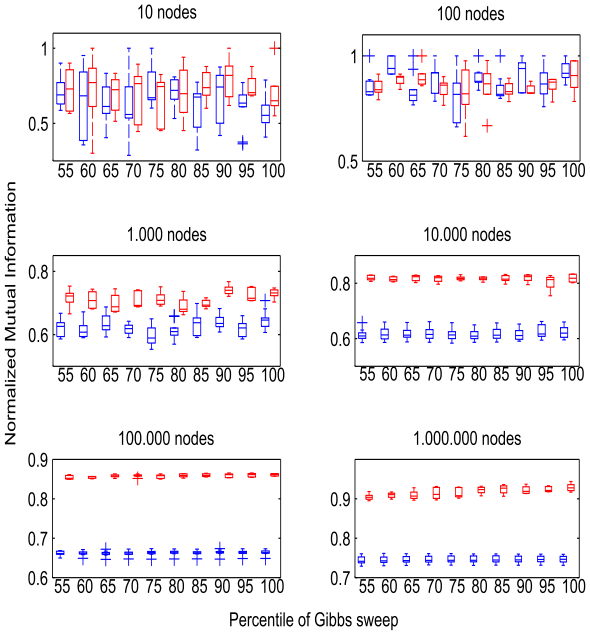


**Fig. 8**. Normalized Mutual Information for different sized networks. (blue) between runs, (red) within runs.

## 6. REFERENCES


[1] K. Börner, S. Sanyal, and A. Vespignani, "Network science," *Annual Review of Information Science & Technology*, vol. 41, pp. 537–607, 2007.

[2] K. Nowicki and T. A. B. Snijders, "Estimation and prediction for stochastic blockstructures," *The American Statistical Association*, vol. 96, no. 455, pp. 1077–1087, 2001.

[3] C. Kemp, J. B. Tenenbaum, T. L. Griffiths, T. Yamada, and N. Ueda, "Learning systems of concepts with an infinite relational model," *Proceedings of the national conference on artificial intelligence*, vol. 21, no. 1, pp. 4–27, 2006.

[4] Z. Xu, V. Tresp, K. Yu, and H.-P. Kriegel, "Learning infinite hidden relational models," *IEEE Signal Processing Magazine*, 2006, in Proceedings of the 22nd International Conference on Uncertainty in Artificial Intelligence.

[5] M. N. Schmidt and M. Mørup, "Non-parametric bayesian modeling of complex networks," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 110–128, 2013.

[6] S. Zhu, K. Yu, and Y. Gong, "Stochastic relational models for large-scale dyadic data using mcmc," *Advances in Neural Information Processing Systems 21 - Proceedings of the 2008 Conference*, pp. 1993–2000, 2009.

[7] T. J. Hansen, M. Mørup, and L. Kai Hansen, "Non-parametric co-clustering of large scale sparse bipartite networks on the gpu," *2011 IEEE International Workshop on achine Learning for Signal Processing*, 2011.

[8] K. Palla, D.A. Knowles, and Z. Ghahramani, "An infinite latent attribute model for network data," *Proceedings of the 29th International Conference on Machine Learning*, pp. 1607–1614, 2012.

[9] K.T. Miller, T.L. Griffiths, and M.I. Jordan, "Nonparametric latent feature models for link prediction," *Advances in Neural Information Processing Systems 22*, pp. 1276–1284, 2009.

[10] P. Hagmann, L. Cammoun, X. Gigandet, R. Meuli, C. J. Honey, V. J. Wedeen, and O. Sporns, "Mapping the structural core of human cerebral cortex," *PLoS biology*, vol. 6, no. 7, pp. 1479–1493, 2008.

[11] S. Jain and R.M. Neal, "A split-merge markov chain monte carlo procedure for the dirichlet process mixture model," *Journal of Computational and Graphical Statistics*, vol. 13, no. 1, pp. 158182, 2004.

[12] M Mørup, K.H. Madsen, A.M. Dogonowski, H. Siebner, and L.K. Hansen, "Innite relational modeling of functional connectivity in resting state fmri," *Neural Information Processing Systems 23*, 2010.

[13] K.W. Andersen, M. Mørup, H. Siebner, Madsen K.H, and L.K. Hansen, "Identifying modular relations in complex brain networks," *MLSP*, 2012.