Flash: Scalable Spatial Probabilistic Graphical Modeling

Ibrahim Sabek
Department of Computer Science and Engineering, University of Minnesota, USA
Email: sabek001@umn.edu

ABSTRACT

The current explosion in spatial data raises the need for efficient spatial analysis tools to extract useful information from such data. Spatial probabilistic graphical modeling (SPGM) is an important class of spatial data analysis that provides efficient probabilistic graphical models for spatial data. Unfortunately, existing SPGM tools are neither generic nor scalable when dealing with big spatial data. In this work, we present *Flash*; a framework for *generic* and *scalable* spatial probabilistic graphical modeling (SPGM). *Flash* exploits Markov Logic Networks (MLN) to express SPGM as a set of declarative logical rules. In addition, it provides spatial variations of the scalable RDBMS-based learning and inference techniques of MLN to efficiently perform SPGM predictions. We have evaluated *Flash*, based on three real spatial analysis applications, and achieved at least two orders of magnitude speed up in learning the modeling parameters over state-of-the-art computational methods.

KEYWORDS

Spatial Probabilistic Graphical Models, Spatial Analysis, Markov Logic Networks, Scalability

1 INTRODUCTION

There is a plethora of spatial data being generated at the moment. For example, space telescopes generate up to 150 gigabytes weekly spatial data, medical devices produce spatial images (X-rays) at a rate of 50 petabytes per year, and a NASA archive of satellite earth images has more than 500 terabytes. This raises the need for efficient spatial analysis solutions to extract insights and useful patterns from such data. *Spatial probabilistic graphical modeling* (SPGM) represents an essential class of spatial analysis techniques, which exploits probability distributions and graphical representations (e.g., spatial hidden Markov models [12]) to describe spatial phenomena and make predictions about them [28]. SPGM has revolutionized many scientific and engineering fields in the past two decades including health care, risk analysis, and environmental science (e.g., [3, 12]). However, existing SPGM techniques have a scalability issue. In particular, they were originally designed for running on a single machine and hence suffer from the limited computation resources (e.g., see [7, 14, 30]). Such techniques can not scale beyond implementing prototypes over small spatial datasets.

Meanwhile, Markov Logic Networks (MLN) [22] was introduced to efficiently build complex learning and inference models over big data in a declarative manner. Basically, MLN combines first-order logic rules with probabilistic graphical models to represent statistical learning and inference problems with few logical rules (e.g., rules with imply and bit-wise AND predicates) instead of thousands of lines of code. With MLN, data scientists and developers can focus their efforts only on developing the rules that represent their applications (e.g., knowledge base construction, data cleaning, genetic analysis). Although the recent advances in MLN frameworks [21] helped to scale up the performance of typical spatial analysis applications (e.g., spatial regression [25, 27], and spatial-aware knowledge base construction [24, 26]), MLN was never exploited to scale up the performance of SPGM techniques.

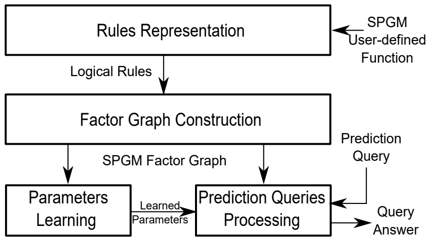


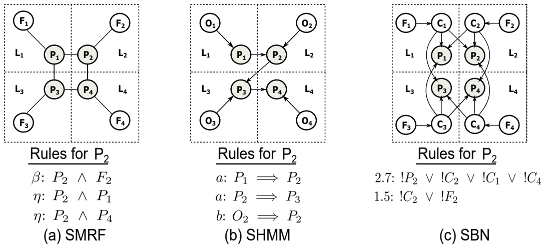
Figure 1: *Flash* System Architecture.

In this paper, we propose *Flash*; a framework for scalable spatial probabilistic graphical modeling (SPGM) using Markov Logic Networks (MLN). *Flash* has the following three main features: (1) *Declarativity*: *Flash* expresses any SPGM application with logical semantics, and allows developers to implement it using a set of logical rules. (2) *Efficiency*: *Flash* translates the equivalent MLN rules of any SPGM application into SQL queries using an efficient grounding technique [29], and then executes these queries inside scalable database engines. In addition, *Flash* provides spatial variations of the RDBMS-based learning and inference algorithms of MLN [21] to perform scalable SPGM predictions (e.g., predictions over models with millions of nodes). (3) *Abstraction*: *Flash* allows developers to build a myriad of spatial analysis applications as a set of user-defined functions (UDF) without the need to worry about the underlying SPGM computation. As a case study, we equipped *Flash* with the implementation of three fundamental SPGMs; spatial Markov random fields (SMRF) [6], spatial hidden Markov models (SHMM) [12], and spatial Bayesian networks (SBN) [7]. The following sections explain the architecture of *Flash*, the implementation details of these three supported SPGMs, and the preliminary evaluation results.

2 FRAMEWORK OVERVIEW

Flash adopts a modular system architecture as shown in Figure 1. It consists of four main modules, described briefly as follows:
Rules Representation. This module is responsible for generating an equivalent representation of logical MLN rules to any user-defined SPGM input. These rules have two main properties: (1) they contain first-order logical predicates (e.g., bitwise-AND, and imply) that capture the SPGM semantics; (2) they are associated with *weights* that represent the original SPGM parameters (Examples

[1] This work is partially supported by the National Science Foundation, USA, under Grants IIS-1907855, IIS-1525953 and CNS-1512877.



Rules for P_2

β: $P_2 \land F_2$

η: $P_2 \land P_1$

η: $P_2 \land P_4$

(a) SMRF

Rules for P_2

a: $P_1 \implies P_2$

a: $P_2 \implies P_3$

b: $O_2 \implies P_2$

(b) SHMM

Rules for P_2

2.7: $!P_2 \lor !C_2 \lor !C_1 \lor !C_4$

1.5: $!C_2 \lor !F_2$

(c) SBN

Figure 2: SMRF, SHMM, and SBN Representations in *Flash*.

are in Section 3). The generated rules follow the syntax of a DBMS-friendly Datalog-like language, called DDlog [29], which can be efficiently processed with any relational DBMS (e.g., PostgreSQL) during the factor graph construction module.

Factor Graph Construction. This module takes the generated rules as input and uses them to build a *factor graph* [33] in a scalable way. The factor graph is the main data structure used to represent any MLN model, where the weights of the graph nodes correspond to the weights of rules (i.e., SPGM model parameters). To efficiently populate this factor graph, *Flash* adapts a scalable grounding technique from [29] that translates the generated rules into SQL queries, and then applies such queries on the input application data to obtain the final factor graph that is equivalent to the SPGM input.

Parameters Learning. This module learns the unknown weights of the constructed factor graph (i.e., weights of rules), which in turn specify the final SPGM parameters (e.g., spatial hidden Markov model [12] parameters). *Flash* proposes a pseudo-likelihood learning algorithm that adapts an efficient variation of a sampling-based gradient descent optimization technique to compute the gradient of the SPGM pseudo-likelihood and then determine the weights.

Prediction Queries Processing. This module is responsible for answering prediction queries over the SPGM model (e.g., what is the probability of a specific event to happen?). Basically, it takes the prediction query along with the factor graph and its learned weights as inputs, and produces a prediction output associated with its confidence probability. Prediction queries can be answered using traditional Gibbs sampling-based inference algorithms over factor graphs [21]. However, such algorithms perform sequential sampling over the factor graph nodes which results in slow convergence to the inference answer in case these nodes have spatial dependencies as in SPGM applications [18]. Instead, *Flash* employs a variation of Gibbs Sampling that exploits a *concliques*-based traversal pattern [18] to efficiently sample spatially-dependent nodes in parallel while guaranteeing the rapid convergence.

3 CASE STUDIES IN *FLASH*

Flash supports the implementation of three common spatial graphical models; spatial Markov random fields (SMRF) [6], spatial hidden Markov models (SHMM) [12], and spatial Bayesian networks (SBN) [7], as case studies. Figure 2 gives toy examples on the logical representation of these three models in *Flash*, where each model is defined over 4-cells grid, and the neighborhood of any cell l is assumed to be the cells that share edges with l only.

Spatial Markov Random Field (SMRF). SMRFs are powerful and important tools for modeling spatial data and building analysis applications. They have been widely used in different areas of spatial statistics [13, 23, 34]. As with many other areas of statistics, a major challenge for spatial analysts is dealing with massive data sets. This is particularly problematic for SMRFs due to the need for matrix operations that involve very large matrices can be computationally prohibitive, specially in the case of Gaussian processes. Existing approaches tried to solve the scalability issues in two categories. The first category focused on developing fast matrix computations that exploit the sparsity of matrices in SMRF models (e.g., [10]). However, utilizing sparsity does not seem to be among the more promising strategies as it does not fit the dense data cases. The second category introduced fast likelihood approximations for the Gaussian-based SMRF models (e.g., [4]). However, this category is not generic enough to capture other arbitrary SMRF models (i.e., the interactions between random variables in the SMRF model are not captured with multivariate Gaussian distributions).

In contrast to existing approaches, *Flash* provides a scalable approach for SMRF models by introducing a first-order logic representation, where there is an equivalent weighted *bitwise-AND* predicate for each pair of connected variables. In this case, the predicates' weights correspond to the SMRF parameters that need to be learned. Figure 2(a) shows a small SMRF model with a prediction P_l and feature F_l at each cell l. Each prediction P_l has undirected edges with feature F_l at this cell and each neighboring prediction variable. For example, predication P_2 is connected with feature F_2 and neighboring predictions P_1 and P_4.

Spatial Hidden Markov Models (SHMM). The hidden Markov model (HMM) is a doubly embedded stochastic method based on probability theory, which can be used in a sequence labeling problem. It describes the process of randomly generating non-observable state sequences from a hidden Markov chain and generating an observation from each state to produce an observable sequence. In spatial hidden Markov model (SHMM), the sequences are generated on spatially-correlated random variables.

While all existing SHMM solutions are innovative, they face severe scalability issues when dealing with big spatial data. The scalability challenge is mainly because these solutions were not originally designed for the big data era or to exploit new high performance computing environments [2]. In contrast, *Flash* scales up the performance of SHMM by providing an equivalent MLN representation, where any state/state or observation/state pair is mapped to a weighted *imply* predicate, and the resulting weights correspond to the SHMM parameters. Figure 2(b) shows a small SHMM model with a hidden state P_l and observation O_l variables at each cell l. Each observation O_l has a directed edge to state P_l at this cell. In addition, SHMM imposes an ordered spatial dependence among neighboring locations, where it uses z-curve ordering technique to build a sequence that preserves the spatial dependence between prediction variables (e.g., P_1 has a directed edge to P_2, and P_2 has another one to P_3, etc).

Spatial Bayesian Networks (SBN). Numerous applications model the probability of an input event to occur based on other *causal* events that have spatial dependencies with the input event. These applications include meteorology [8], risk analysis [3, 17], and environmental science [11, 19]. For example, business analysts forecast the budget and likely costs of water infrastructure networks based on failure events in water mains at neighboring sites [17]. A typical solution to model the *causal dependencies* between events in all these applications is to employ spatial Bayesian networks (a.k.a spatial Bayesian belief networks) [16, 20]. These networks are directed probabilistic graphs whose nodes represent variables corresponding to events over neighboring locations, and the edges represent the casual relationships between these variables. For example, two events "rain" and "flood" at neighboring locations x and y, respectively, can be represented as two random variables, where the *rain* variable is a cause (i.e., parent node in the graph) for the *flood* variable. Existing solutions of spatial Bayesian networks can not scale beyond implementing prototypes over small spatial and spatio-temporal datasets [7, 20]. Meanwhile, Markov Logic Networks (MLN) are recently used to scale up the performance of classical Bayesian networks that do not consider spatial dependencies between random variables (e.g., Bayesian Logic Networks [15]).

In *Flash*, we exploit Markov Logic Networks (MLN) to represent the SBN models. *Flash* provides an equivalent weighted combination of *bitwise-OR* and *negation* predicates for each causality relation (i.e., directed edge). The weights of these predicates are calculated from the input prior probabilities of SBN. Figure 2(c) shows a small SBN model with a prediction variable P_l at each cell l which is affected directly by a status variable C_l and indirectly by a feature variable F_l (i.e., F_l has a direct edge to C_l). In addition, each prediction P_l is affected by the status variables at the neighboring cells.

4 EXPERIMENTS

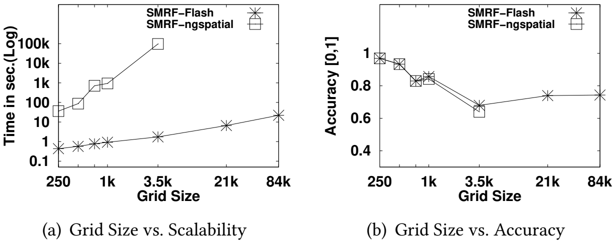
In this section, we experimentally evaluate the accuracy and scalability of *Flash* in building SPGM models for three spatial analysis applications. In these applications, we compare the performance of *Flash* with ngspatial [14], shmm [30], and bnspatial [7] tools when building SMRF [6], SHMM [12], and SBN [7] models, respectively.

4.1 Experimental Setup

Applications. The details of the three applications, along with their datasets, used in our experiments are described as follows:

Bird Monitoring. This application predicts the existence of a bird species across a certain area. Ornithologists model this problem using SMRF [14] as shown in [1], where the area is divided by a two-dimensional grid. Each grid cell holds a binary prediction variable indicating the presence or absence of the bird at this cell, and a set of feature variables that help predicting the value of this prediction variable. Then, the prediction at any cell is determined based on the values of feature variables at this cell along with a set of predicted or observed values at neighbouring cells. As a case study, we use the daily distribution of a certain bird species, namely Barn Swallow, from Ebird dataset [31], which contains more than 360 Million observations collected over North America. We define a grid of 84K cells, and map each observation to one cell. Then, we build the SMRF-based prediction model of the bird existence at cells with no observations.

Safety Analysis. The objective of this application is to infer the safety level (e.g., low, medium and high) at a bunch of neighbouring locations simultaneously based on reported incidents at these locations. This application has been usually represented with SHMM [12] as shown in [5], where the safety level at each location is considered a hidden state to be predicted and the reported incident at this location is an observation that affects the prediction value. As a case study, we use the official Chicago crime dataset repository [9], which contains around 7 Million reported incidents (i.e., observations) over 500K grid locations.



(a) Grid Size vs. Scalability

(b) Grid Size vs. Accuracy

Figure 3: Study of SMRF Model Scalability and Accuracy.

Land Use Change Tracking. The objective of this application is to determine whether there will be a change in the land use or not. For example, the land in a location l could be suitable for agriculture, however, given certain factors (e.g., crowded neighbourhoods), it is expected to be for human use soon. We model this application as SBN problem. As a case study, we use a grid dataset containing one Million cells of land cover distribution over Minnesota state, and is compiled from national land cover data repository [32].

In each application, we randomly select 15% of its grid data for testing, and use the rest data for training any SPGM model.

Environment. We run all experiments on a single machine with Ubuntu Linux 14.04. Each machine has 8 quad-core 3.00 GHz processors, 64GB RAM, and 4TB hard disk.

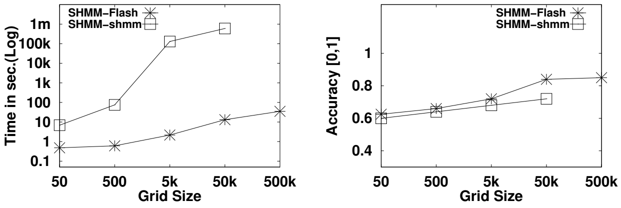
Metrics. We use the total running time of learning the parameters of any SPGM model as a scalability evaluation metric, and the ratio of correctly predicted cells using the learned model to the total number of test cells as an accuracy evaluation metric.

4.2 Experimental Results

4.2.1 Study of SMRF Scalability and Accuracy. In this section, we compare the performance, both scalability and accuracy, of *Flash* with ngspatial [14], when learning and using the SMRF models that are built for five different sizes of Ebird grid data.

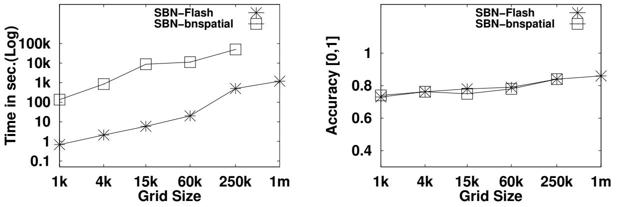
Figure 3(a) shows the running time for each algorithm to learn the SMRF parameters while scaling the grid size from 250 to 84k cells. For all sizes, *Flash* was able to significantly reduce the running time compared to ngspatial. Specifically, *Flash* and ngspatial have an average running time of 4.7 seconds and 5.5 hours, respectively. This means that *Flash* has at least three orders of magnitude reduction in the running time over ngspatial. Note that the ngspatial curve in Figure 3(a) is incomplete after a grid size of 3.5k cells because of a failure in satisfying the memory requirements needed for its internal computations. In contrast, the running times for *Flash* are complete. This shows the *Flash* efficiency when scaling up the grid size regardless of the model specified.

Figure 3(b) shows the accuracy for each algorithm while using the same grid sizes in Figure 3(a). As can be seen in the figure, *Flash* has almost the same accuracy achieved by ngspatial at small grid sizes, while it is slightly more accurate (4% more) than ngspatial at the grid size of 3.5k cells. Note that the ngspatial curve is incomplete for grids with sizes more than 3.5k cells as in Figure 3(a).



(a) Grid Size vs. Scalability (b) Grid Size vs. Accuracy

Figure 4: Study of SHMM Model Scalability and Accuracy.



(a) Grid Size vs. Scalability (b) Grid Size vs. Accuracy

Figure 5: Study of SBN Model Scalability and Accuracy.

4.2.2 Study of SHMM Scalability and Accuracy. In this section, we compare the performance, both scalability and accuracy, of *Flash* with shmm [30], when learning and using the SHMM models that are built for five different sizes of Chicago crime grid data.

Figure 4(a) shows the running time for each algorithm while scaling the grid size from 50 to 500k cells. We can observe from the results that *Flash* has an average three orders of magnitude less running time than shmm. In contrast to ngspatial in Figure 3(a), shmm is more scalable to relatively large grid sizes (e.g., 50k cells), but, still can not complete the running for huge sizes like 500k cells.

Figure 4(b) shows the prediction accuracy for each algorithm while using the same grid sizes in Figure 4(a). We observe that *Flash* is consistently more accurate than shmm at all sizes, yet, *Flash* has a larger improvement ratio when the grid size becomes larger (the improvement ratio can reach to 18%). Note that the shmm curve is also incomplete for the grid with 500k cells as in Figure 4(a).

4.2.3 Study of SBN Scalability and Accuracy. In this section, we compare the performance, both scalability and accuracy, of *Flash* with bnspatial [7], when learning and using the SBN models that are built for six different sizes of Minnesota land use data.

Figure 5(a) shows the running time for each algorithm while scaling the grid size from 1k to 1 million cells. In general, *Flash* is much faster than bnspatial in all cases, however the ratio of improvement in case of SBN is less than its counterpart in SHMM. We observe from the results that *Flash* has at least two orders of magnitude less running times than bnspatial. The running times of *Flash* range from 0.6 sec (min. value) to 20 hours (max. value).

Figure 5(b) shows the accuracy for each algorithm while using the same grid sizes in Figure 5(a). As shown, *Flash* does not improve so much over the accuracy already obtained by bnspatial. In general, the main objective of *Flash* is to speed up the computation steps of SPGM models, while keeping the same accuracy obtained by the best state-of-the-art techniques or increasing it, if possible.

REFERENCES

[1] R. Ambrosini et al. Modelling the Progression of Bird Migration with Conditional Autoregressive Models Applied to Ringing Data. *PLOS ONE*, 9(7):1–10, 2014.

[2] L. Anselin et al. Spatial Econometrics in an Age of CyberGIScience. *IJGIS*, 2012.

[3] S. Balbi et al. A Spatial Bayesian Network Model to Assess the Benefits of Early Warning for Urban Flood Risk to People. *Natural Hazards and Earth System Sciences*, 2016.

[4] S. Banerjee, A. Gelfand, A. Finley, and H. Sang. Gaussian Predictive Process Models for Large Spatial Data Sets. *Royal Statistical Society Journal*, 70(4), 2008.

[5] F. Bartolucci et al. A Latent Markov Model for Detecting Patterns of Criminal Activity. *Royal Statistical Society Journal*, 170(1):115–132, 2007.

[6] J. Besag. Spatial Interaction and the Statistical Analysis of Lattice Systems. *Royal Statistical Society Journal*, 1974.

[7] bnspatial: Spatial Implementation of Bayesian Networks. cran.r-project.org/web/packages/bnspatial, 2019.

[8] R. Cano, C. Sordo, and J. M. Gutiérrez. *Applications of Bayesian Networks in Meteorology*, pages 309–328. Springer, 2004.

[9] Chicago Crime Data, 2019. data.cityofchicago.org/Public-Safety/Crimes-2001-to-present/ijzp-q8t2/.

[10] D. Cornford, L. Csató, and M. Opper. Sequential, Bayesian Geostatistics: A Principled Method for Large Data Sets. *Geographical Analysis*, 37(2), 2005.

[11] J. Gonzalez-Redin et al. Spatial Bayesian Belief Networks as a Planning Decision Tool for Mapping Ecosystem Services Trade-offs on Forested Landscapes. *Environmental Research*, pages 15–26, 2016.

[12] P. J. Green and S. Richardson. Hidden Markov Models and Disease Mapping. *JASA*, pages 1055–1070, 2002.

[13] M. Haran. *Gaussian Random Field Models for Spatial Data*, pages 449–478. Chapman and Hall/CRC, 2011.

[14] J. Hughes. ngspatial: A Package for Fitting the Centered Autologistic and Sparse Spatial Generalized Linear Mixed Models for Areal Data. *The R Journal*, 2014.

[15] D. Jain, K. von Gleissenthall, and M. Beetz. Bayesian Logic Networks and the Search for Samples with Backward Simulation and Abstract Constraint Learning. In *KI 2011: Advances in Artificial Intelligence*, 2011.

[16] F. V. Jensen. *Introduction to Bayesian Networks*. Springer-Verlag, 1996.

[17] G. Kabir, S. Tesfamariam, A. Francisque, and R. Sadiq. Evaluating Risk of Water Mains Failure using a Bayesian Belief Network Model. *European Journal of Operational Research*, 2015.

[18] M. Kaiser et al. Goodness of Fit Tests for a Class of Markov Random Field Models. *The Annals of Statistics*, pages 104–130, 2012.

[19] D. Landuyt, S. Broekx, R. D'hondt, G. Engelen, J. Aertsens, and P. L. M. Goethals. A Review of Bayesian Belief Networks in Ecosystem Service Modelling. *Environmental Modelling and Software*, 2013.

[20] M. Mello, B. Rudorff, M. Adami, and D. Aguiar. An R Implementation for Bayesian Networks Applied to Spatial Data. *Procedia Environmental Sciences*, 2011.

[21] F. Niu et al. Tuffy: Scaling Up Statistical Inference in Markov Logic Networks Using an RDBMS. *PVLDB*, 4(6):373–384, 2011.

[22] M. Richardson and P. M. Domingos. Markov Logic Networks. *Machine Learning*, pages 107–136, 2006.

[23] H. Rue and L. Held. *Gaussian Markov Random Fields: Theory And Applications (Monographs on Statistics and Applied Probability)*. Chapman & Hall/CRC, 2005.

[24] I. Sabek and M. F. Mokbel. Sya: Enabling Spatial Awareness inside Probabilistic Knowledge Base Construction. In *ICDE*, 2020.

[25] I. Sabek, M. Musleh, and M. Mokbel. TurboReg: A Framework for Scaling Up Spatial Logistic Regression Models. In *SIGSPATIAL*, pages 129–138, 2018.

[26] I. Sabek, M. Musleh, and M. F. Mokbel. A Demonstration of Sya: A Spatial Probabilistic Knowledge Base Construction System. In *SIGMOD*, 2018.

[27] I. Sabek, M. Musleh, and M. F. Mokbel. RegRocket: Scalable Multinomial Autologistic Regression with Unordered Categorical Variables Using Markov Logic Networks. *ACM TSAS*, 5(4):27:1–27:27, 2019.

[28] S. Shekhar et al. Identifying Patterns in Spatial Information: A Survey of Methods. *WIRES: Data Mining and Knowledge Discovery*, pages 193–214, 2011.

[29] J. Shin et al. Incremental Knowledge Base Construction Using DeepDive. *PVLDB*, 8(11):1310–1321, 2015.

[30] shmm: An R Implementation of Spatial Hidden Markov Models. github.com/mawp/shmm, 2019.

[31] B. Sullivan et al. eBird: A Citizen-based Bird Observation Network in the Biological Sciences. *Biological Conservation*, pages 2282–2292, 2009.

[32] USGS National Land Cover. archive.usgs.gov/archive/sites/landcover.usgs.gov/landcoverdata.html, 2019.

[33] M. Wick et al. Scalable Probabilistic Databases with Factor Graphs and MCMC. *PVLDB*, 3(1-2):794–804, 2010.

[34] J. Zhu, H.-C. Huang, and J. Wu. Modeling Spatial-temporal Binary Data using Markov Random Fields. *JABES*, 2005.