Probability in the Engineering and Informational Sciences, **32**, 2018, 615–625.
doi:10.1017/S0269964817000432

A BIRTH AND DEATH PROCESS FOR BAYESIAN NETWORK STRUCTURE INFERENCE

DALE JENNINGS AND JEM N. CORCORAN
Applied Mathematics, University of Colorado, 526 UCB, Boulder, Colorado, USA
E-mail: corcoran@colorado.edu

Bayesian networks are convenient graphical expressions for high-dimensional probability distributions which represent complex relationships between a large number of random variables. They have been employed extensively in areas such as bioinformatics, artificial intelligence, diagnosis, and risk management. The recovery of the structure of a network from data is of prime importance for the purposes of modeling, analysis, and prediction. There has been a great deal of interest in recent years in the NP-hard problem of learning the structure of a Bayesian network from observed data. Typically, one assigns a score to various structures and the search becomes an optimization problem that can be approached with either deterministic or stochastic methods. In this paper, we introduce a new search strategy where one walks through the space of graphs by modeling the appearance and disappearance of edges as a birth and death process. We compare our novel approach with the popular Metropolis–Hastings search strategy and give empirical evidence that the birth and death process has superior mixing properties.

Keywords: Bayesian networks, birth and death processes, structure learning

1. INTRODUCTION

Bayesian networks are convenient graphical expressions for high-dimensional probability distributions representing complex relationships between a large number of random variables. They have been employed extensively in areas such as bioinformatics [4,12,21,22], artificial intelligence [13,19,27], diagnosis [7,24,29], and risk management [6,10,28] for the purposes of general modeling, prediction, and diagnosis.

A Bayesian network consists of a directed acyclic graph (DAG), in which nodes represent random variables, and arrows correspond to a set of conditional probability distributions. For example, the graph in Figure 1 represents a network for five variables, X_1, X_2, X_3, X_4, and X_5 with a joint probability density $p(x_1, x_2, x_3, x_4, x_5)$ which is assumed to factor as

$$p(x_1, x_2, x_3, x_4, x_5) = p(x_1) \cdot p(x_2|x_1) \cdot p(x_3|x_1) \cdot p(x_4|x_2, x_3) \cdot p(x_5|x_4).$$

© Cambridge University Press 2017 0269-9648/17 $25.00

https://doi.org/10.1017/S0269964817000432 Published online by Cambridge University Press

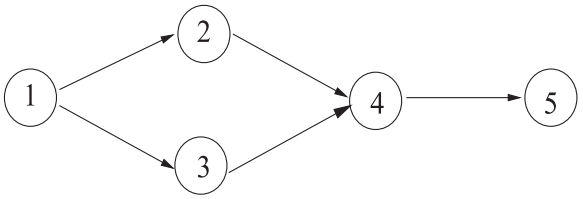


FIGURE **1.** A DAG.

In general, for a Bayesian network on N nodes, the assumption is that joint density for the corresponding N random variables may be written as

$$p(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{N} p(x_i | pa_i) \tag{1}$$

where pa_i is used to denote the set of random variables corresponding to "parent" nodes for node i. For example in Figure 1, we have $pa_1 = \emptyset$, $pa_2 = \{x_1\}$, $pa_3 = \{x_1\}$, $pa_4 = \{x_2, x_3\}$, and $pa_5 = \{x_4\}$.

Given data consisting of M observations of $(X_1, X_2, \ldots, X_N)$, our ultimate goal is to infer the connecting arrows for a corresponding N-node graph. There has been a great deal of interest in recent years [5,8,11,12,14,16–18,23] on this NP-hard problem. Such structure learning often involves defining a network "score function" that will indicate how well a particular DAG fits the data. Once the score function is defined, the problem of finding the best-fitting DAG becomes, in theory, a straightforward optimization problem. In practice, however, it can be be extremely difficult as the number of possible networks to be scored and compared grows super-exponentially with the number of nodes. Simple greedy hill climbing with random restarts is inefficient yet surprisingly hard to beat. There have been many deterministic and stochastic alternatives proposed in recent years, including gradient descent, genetic, simulated tempering, and Metropolis–Hastings algorithms. Several of these methods have offered some improvement over greedy hill climbing but can be difficult to implement. Deterministic methods tend to get stuck in local maxima and probabilistic methods tend to suffer from slow mixing.

In this paper, we consider a new stochastic algorithm in which the appearance and disappearance of edges are modeled as a birth and death process. We compare our algorithm with the popular Metropolis–Hastings algorithm and give empirical evidence that ours has better mixing properties.

2. NOTATION AND A TARGET

A Bayesian network on N nodes is a DAG where the nodes (vertices), labeled $1, 2, \ldots, N$, correspond to random variables $X_1, X_2, \ldots, X_N$ and directed edges correspond to probabilistic dependencies between them. We say that node i is a *parent* of node j and node j is a *child* of node i if there exists a directed edge from i to j, (in which case we write $i \to j$), and use the notation pa_j to denote the collection of all parents of node j. We will refer to nodes and their associated random variables interchangeably. Thus, pa_j may also represent the collection of parent random variables for X_j. Rigorously, a Bayesian network consists

https://doi.org/10.1017/S0269964817000432 Published online by Cambridge University Press

of a DAG and a set of conditional densities $\{p(x_i|pa_i)\}_{i=1}^N$ along with the assumption that the joint density for the N random variables can be written as the product

$$p(x_1, x_2, \ldots, x_N) = \prod_{i=1}^{N} p(x_i|pa_i).$$

In other words, all nodes are conditionally independent given their parents.

In the problem of structure inference, the DAG is not explicitly known. A data set $D = (D_1, D_2, \ldots, D_M)$ is given, where each D_i is an N-tuple realization of $(X_1, X_2, \ldots, X_N)$. The goal is then to recover the "best" Bayesian network structure (graph) G from the space of all possible DAGs on N nodes. "Best" may be measured in many ways. For example, one may consider the best DAG as the one that maximizes the posterior probability

$$P(G|D) \propto P(D|G)P(G) \tag{2}$$

over G. Here, $P(D|G)$ is the likelihood for the full data set and $P(G)$ is a prior probability on DAGs that is often taken to be uniform over the space of DAGs.

Not surprisingly, $P(D|G)$ will favor DAGs with a maximal number of edges. In practice, one would include penalty terms for overfitting, for example using Akaike's information criterion [1]. However, as Eq. (2) is central to many measures of "best DAGs", it is the goal of this paper to simulate DAGs efficiently from this distribution.

3. THE JUMP PROCESS

Let $(\Omega, \mathcal{F})$ be a state space consisting of a non-empty set Ω and a sigma algebra $\mathcal{F}$ on Ω. A *jump process* on $(\Omega, \mathcal{F})$ is a continuous time stochastic process that is characterized as follows. Assume that the process is currently in some state $x \in \Omega$.

- The waiting time until the next jump follows an exponential distribution with rate (or intensity) $\lambda(x)$ and is independent of the past history.
- The probability that the jump lands the process in $F \in \mathcal{F}$ is given by a transition kernel $K(x, F)$.

It is known (e.g. [9,25]) that there exists a $P_t : \Omega \times \mathcal{F} \to \mathbb{R}^+$ so that $P_t(x, F)$ is the probability that at time t the process is in F given that the process was in state x at time 0. Such P_t are defined as the solution to the Kolmogorov backward equation

$$\frac{\partial}{\partial t} P_t(x, F) = -\lambda(x) P_t(x, F) + \lambda(x) \int_\Omega P_t(y, F) K(x, dy).$$

Let $P_t^{(n)}(x, F)$ denote the probability of a transition from x to F using at most n jumps. If $\lambda(x)$ is bounded then, it is furthermore known that

$$P_t^{(\infty)}(x, F) := \lim_{n \to \infty} P_t^{(n)}(x, F)$$

is the unique minimal solution to the Kolmogorov forward equation

$$\frac{\partial}{\partial t} P_t(x, F) = -\int_F \lambda(z) P_t(x, dz) + \int_\Omega \lambda(z) K(z, F) P_t(x, dz).$$

(It is "minimal" in the sense that if $R_t(x, F)$ is any other non-negative solution, then $R_t(x, F) \geq P_t^{(\infty)}(x, F)$ for all $t \geq 0$, $x \in \Omega$, and $F \in \mathcal{F}$.)

https://doi.org/10.1017/S0269964817000432 Published online by Cambridge University Press

For a distribution, π, to be invariant (stationary) under such a process, it is sufficient that π satisfy the detailed balance condition

$$\pi(x)\lambda(x)K(x,dy) = \pi(y)\lambda(y)K(y,dx).$$

In Section 4, we will impose this condition in order to construct appropriate transition probabilities in order to target an invariant distribution of interest.

The state space Ω has some dimension n. We will emphasize this with the notation Ω_n in place of Ω. Preston [25] extended the jump process to the trans-dimensional case where jumps from states in Ω_n can move the process to a state in Ω_{n+1}, with (birth) rate $\lambda_{\mathrm{b}}(x)$ or to a state in Ω_{n-1}, with (death) rate $\lambda_{\mathrm{d}}(x)$. Associated with these birth and death rates are birth and death kernels, K_{b} and K_{d}. The total jump rate and the transition kernel are then given by

$$\lambda(x) = \lambda_{\mathrm{b}}(x) + \lambda_{\mathrm{d}}(x),$$

and

$$K(x,F) = \frac{\lambda_{\mathrm{b}}(x)}{\lambda(x)}K_{\mathrm{b}}(x,F) + \frac{\lambda_{\mathrm{d}}(x)}{\lambda(x)}K_{\mathrm{d}}(x,F).$$

We can think of the n-dimensional state $x \in \Omega_n$ as a spatial configuration of n points. For the process in state x with n points to move to a configuration x' with $n+1$ points (or vice versa), the detailed balance conditions simplify [25,26]. In this case, one needs the birth rates to balance the death rates with respect to π. That is, we require that

$$\pi(x)b(x, x' \setminus x) = \pi(x')d(x', x' \setminus x),$$

where $b(x, x' \setminus x)$ is the birth rate of the single point $x' \setminus x$ given that the current configuration of points is x, and $d(x', x' \setminus x)$ is the death rate of the single point $x' \setminus x$ given that the current configuration of points is x'. These relate to the total birth and death rates in that

$$\lambda_{\mathrm{b}}(x) = \sum b(x, x' \setminus x),$$
$$\lambda_{\mathrm{d}}(x') = \sum d(x', x' \setminus x),$$

where the birth sum is taken over all states x' that consist of configuration x with the addition of a single point and the death sum is taken over all states x that consist of configuration x' with a single point deleted.

4. A BIRTH AND DEATH-TYPE PROCESS ON EDGES OF A BAYESIAN NETWORK

To construct a jump process for Bayesian network structure inference, our goal is to construct a birth and death-type process (formally, a quasi-birth and death process) acting on edges of a network which has invariant distribution $P(G|D)$ given by Eq. (2). We will construct a Markov chain that follows the evolution of a DAG through the addition (births) and deletion (deaths) of directed edges.

The relevant state space is $(\mathcal{G}, 2^{\mathcal{G}})$, where $\mathcal{G}$ is the set of all DAGs with N nodes and $2^{\mathcal{G}}$ is the power set of $\mathcal{G}$. We define disjoint sets $\mathcal{G}_k$, $k = 0, \ldots, ((N(N-1))/2)$, to be the set of DAGs with exactly k edges. Our jump process will then jump between the $\mathcal{G}_k$ for adjacent values of k.

https://doi.org/10.1017/S0269964817000432 Published online by Cambridge University Press

For $G \in \mathcal{G}_k$, denote the graph with the addition of the edge from node i to node j by $(G \cup \{i \to j\}) \in \mathcal{G}_{k+1}$, and the graph with the removal of the edge from i to j by $(G \setminus \{i \to j\}) \in \mathcal{G}_{k-1}$. Detailed balance then requires that, for every edge $i \to j$ that is a valid (non-cycle causing) addition,

$$P(G|D)\, b(G, \{i \to j\}) = P(G \cup \{i \to j\}|D)\, d(G \cup \{i \to j\}, \{i \to j\}).$$

It is convenient to let

$$d(G \cup \{i \to j\}, \{i \to j\}) = 1$$

so that

$$\begin{aligned} b(G, \{i \to j\}) &= \frac{P(G \cup \{i \to j\}|D)}{P(G|D)} \\ &= \frac{P(D|G \cup \{i \to j\})P(G \cup \{i \to j\})}{P(D|G)P(G)}. \end{aligned}$$

If we let Δ_j denote the M-dimensional vector of observations of X_j in the data set D, this birth rate may be rewritten as

$$b(G, \{i \to j\}) = \frac{P(\Delta_j|\Delta_{pa'_j}, G \cup \{i \to j\})P(G \cup \{i \to j\})}{P(\Delta_j|\Delta_{pa_j}, G)P(G)}.$$

Here, Δ_{pa_j} is the $M \times r$ matrix of data points for the r parents of node j in G ($\Delta_{pa_j} = \emptyset$ if $r = 0$) and $\Delta_{pa'_j}$ is the $M \times (r+1)$ matrix of data points for the $r+1$ parents of node j in $G \cup \{i \to j\}$.

The transition rates are then given by

$$\begin{aligned} \lambda_{\mathrm{b}}(G) &= \sum_{\text{valid } i \to j} b(G, \{i \to j\}), \\ \lambda_{\mathrm{d}}(G) &= \sum_{\{i \to j\} \in G} 1. \end{aligned}$$

With this, we can easily construct a way to simulate from this process in the following way.

1. Start with an arbitrary initial DAG, G.
2. Compute the birth rates $b(G, \{i \to j\})$ for all possible valid edge additions $i \to j$ in G. Compute
$$\lambda_{\mathrm{b}}(G) = \sum_{\text{valid } i \to j} b(G, \{i \to j\})$$
and
$$\lambda_{\mathrm{d}}(G) = \sum_{\{i \to j\} \in G} 1.$$
3. With probability $\lambda_{\mathrm{d}}(G)/(\lambda_{\mathrm{b}}(G) + \lambda_{\mathrm{d}}(G))$, remove a randomly selected existing edge. Otherwise, add valid edge $i \to j$ with probability $b(G, \{i \to j\})/\lambda_{\mathrm{b}}(G)$.
4. Return to Step 2.

https://doi.org/10.1017/S0269964817000432 Published online by Cambridge University Press

At first glance, it may seem like such an algorithm would be computationally expensive, as the required birth rates depend on computing the score for two different graphs. However, if we make the common assumption of a *modular* likelihood,

$$P(D|G) = \prod_{i=1}^{N} P(\Delta_i|\Delta_{pa_i}, G),$$

the computation at each step is manageable.

This assumption leads to a birth rate of

$$\begin{aligned} b(G, \{i \to j\}) &= \frac{P(G \cup \{i \to j\}) \prod_{i=1}^{N} P(\Delta_i|\Delta_{pa'_i}, G \cup \{i \to j\})}{P(G) \prod_{i=1}^{N} P(\Delta_i|\Delta_{pa'_i}, G)} \\ &= \frac{P(G \cup \{i \to j\}) P(\Delta_j|\Delta_{pa'_j}, G \cup \{i \to j\})}{P(G) P(\Delta_j|\Delta_{pa_j}, G)}, \end{aligned} \tag{3}$$

and so we only need to pay attention to a single node corresponding to the child of the proposed edge.

Further computational relief comes from the fact that most of the birth rates can be stored from previous steps. After adding or removing an edge, the only birth rates that need to be recalculated are for edges pointing to nodes whose parent set has been altered, edges which were not previously valid, or edges which are no longer valid.

These realizations allow for a more complete and efficient algorithm:

1. Start with an initial DAG, G.
2. For all possible edge additions, $i \to j$, calculate the birth rate $b(G, \{i \to j\})$ using Eq. (3).
3. With probability $\lambda_{\mathrm{d}}(G)/(\lambda_{\mathrm{b}}(G) + \lambda_{\mathrm{d}}(G))$, remove a randomly selected existing edge. Otherwise, add a valid edge $k \to \ell$ with probability $b(G, \{k \to \ell\})$.
4. If any of the following are true, update the birth rate $b(G, \{i \to j\})$.
   - $j = \ell$
   - Edge $i \to j$ was not a valid addition before the addition or removal of edge $k \to \ell$ but now is valid
   - Edge $i \to j$ was a valid addition before the addition or removal of edge $k \to \ell$ but now is not longer valid
5. Return to Step 3.

5. EXPERIMENTAL RESULTS

While the theory guarantees that our edge birth and death process will have the correct stationary distribution, in this section we investigate the mixing time and whether or not Monte Carlo simulation of graphs from $P(G|D)$ using our method is feasible in practice.

We will use the common assumption that the data D come from a multinomial distribution, allowing us to analytically integrate out parameters to obtain a score which is proportional to $P(G|D)$. Specifically, we will assume that the random variable X_i, associated

https://doi.org/10.1017/S0269964817000432 Published online by Cambridge University Press

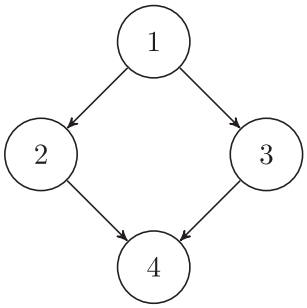


FIGURE **2.** Four-node graph used for testing.

with node i, can take on values $\{1, 2, \ldots, r_i\}$ where r_i is a positive integer. We express the dependency of each variable X_i on its parents as

$$P(X_i = k|pa_i) = \theta_{i,pa_i,k}$$

where pa_i is a particular configuration of the parent variables of X_i.

If we enumerate the number of possible configurations of values taken on by the parent nodes of X_i as $1, 2, \ldots, q_i$ where $q_i = \prod_{j \in pa_i} r_j$, then we may write

$$P(X_i = k|pa_i = j) = \theta_{i,j,k} \tag{4}$$

for

$$i = 1, 2, \ldots, n, \quad j = 1, 2, \ldots, q_i, \quad \text{and} \quad k = 1, 2, \ldots, r_i.$$

Now consider data consisting of M observations of the N nodes of a network, and restrict attention for a moment to the M values of the ith node. Consider any one configuration j of the parent nodes to node i that exists in the data. Let M_j be the number of times that the parents of node i take on configuration j in the data set. Then within the M_j values of $X_i|pa_i = j$, we can describe the number of observed 1's, 2's, and so on, up to the number of r_i's with a multinomial distribution with parameters M_j, r_i, and $(\theta_{i,j,1}, \theta_{i,j,2}, \ldots, \theta_{i,j,r_i})$.

To first test our algorithm to find the DAG G that maximizes $P(G|D)$ in the case where exact values can be computed, we first consider a simple four-node graph, depicted in Figure 2. Setting $r_1 = r_2 = r_3 = r_4 = 4$, we generated 50 observations using uniform probabilities for Eq. (4). That is, parentless node X_1 took on values in $\{1, 2, 3, 4\}$ with probability 1/4 each. Given that node X_1 took on, for example, the value 3, node X_2 would then have taken on values in $\{1, 2, 3, 4\}$ with probability 1/4 each. In general, $\theta_{i,j,k} = 1/4$ for $k \in \{1, 2, 3, 4\}$ and for all i and j.

We then ran both the standard Metropolis–Hastings chain for Bayesian networks [20] and our edge birth and death algorithm. Figure 3 shows 543 points, representing the 543 possible DAGs on four nodes. Though difficult to discern for all but a few DAGs, the sizes of the points in the figure are proportional to the posterior probability of the DAG given the data. Note that for only 50 simulated observations in D, the DAG that maximizes $P(G|D)$ is not necessarily the DAG that generated the data. Thus, the largest point (circle) in Figure 3 does not correspond to the DAG in Figure 2 but rather to the DAG G that maximizes $P(G|D)$. Indeed, even for larger data sets, finding DAGs that maximize this quantity is not enough for network recovery. However, the task is essential to most strategies which would then involve more steps such as penalizing for overfitting edges and distinguishing

https://doi.org/10.1017/S0269964817000432 Published online by Cambridge University Press

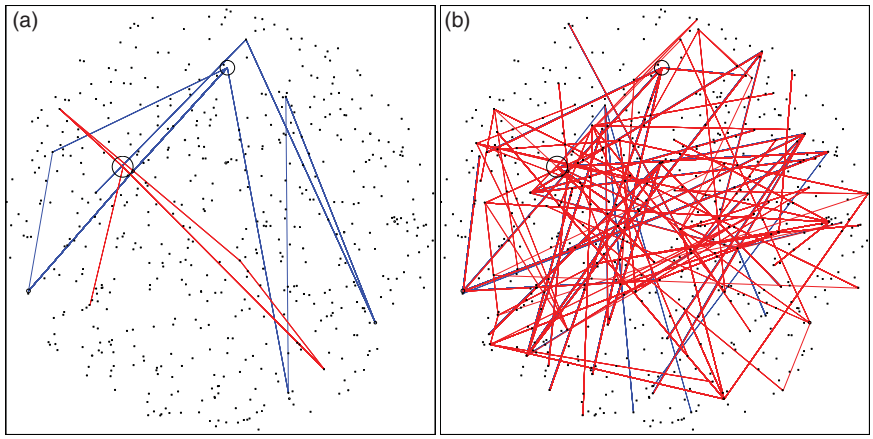


FIGURE **3.** (Color online) (a) Paths of two (red and blue) independent Metropolis–Hastings chains of 20,000 steps. (b) Paths of two independent edge birth and death process runs of 1500 jumps. The circles represent each of the 543 possible DAGs, with size proportional to the posterior probability of the graph.

TABLE 1. Estimated absolute differences between simulated and exact probabilities of edges in a four-node Bayesian network based on 1000 runs of the edge birth and death algorithm for data sets consisting of 100 (left) and 500 (right) observations. Standard errors were of the order of 3×10^{-2} for $M = 100$ and 5×10^{-3} for $M = 500$.

Node	1	2	3	4
$M = 100$				
1	–	0.0525	0.0233	0.0517
2	0.0464	–	0.009	0.011
3	0.0218	0.0464	–	0.013
4	0.0484	0.046	0.010	–
$M = 500$				
1	–	0.021	0.018	0.003
2	0.026	–	0.006	0.002
3	0.021	0.001	–	0.004
4	0.003	0.002	0.001	–

between "Markov equivalent" graphs (see, for example, [14]) that encode the same sets of conditional dependencies.

Figure 3(b) shows two independent runs of each algorithm traced out as red and blue paths.

While both algorithms tend to find high probability regions of the state space, our edge birth and death algorithm clearly explores much more of the state space.

Simulating the edge birth and death process allows us to easily estimate probabilities of specific graph features. For example, edge probabilities correspond to the proportion of time that the graphs contain the given edge. With the same four-node graph as before, we calculated the edge probabilities and compared them with the true edge probabilities for both $M = 100$ and $M = 500$ observations. The results are shown in Table 1.

https://doi.org/10.1017/S0269964817000432 Published online by Cambridge University Press

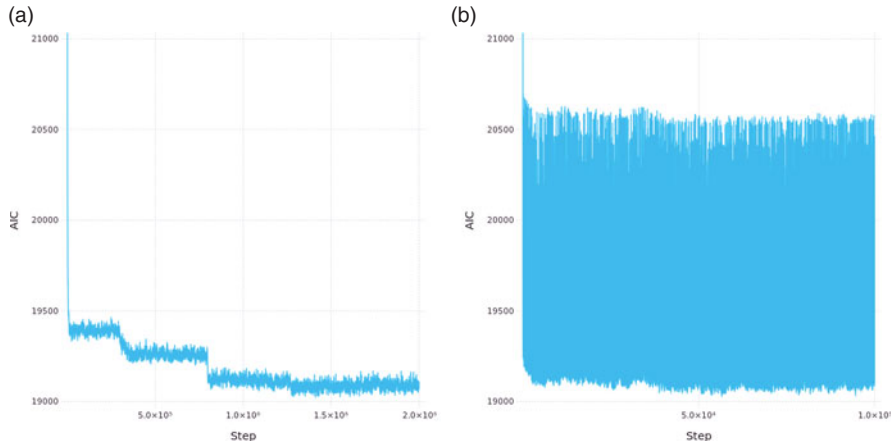


FIGURE **4.** (Color online) (a) AIC score of a Metropolis–Hastings run of 2×10^6 steps. (b) AIC score of an edge birth and death process run of 10^5 steps.

Next, we tested our edge birth and death algorithm on the "Alarm data set" compiled by Herskovits [15]. This data set, often used as a benchmark for structure learning algorithms, consists of 1000 observations of 37 random variables, each taking on four possible values. We ran our edge birth and death algorithm for 10^5 time steps. For comparison, we ran the standard Metropolis–Hastings scheme for 2×10^6 steps which took approximately the same amount of CPU time. Akaike's information criteria (AIC), a log-likelihood score defined for a graph G as

$$AIC = -2 \ln P(G|D) + 2 \cdot (\#\theta \text{ parameters in graph } G),$$

that penalizes for model overparameterization, was recorded at each step and is plotted for one of these runs in Figure 4. When using the AIC score, the goal is to find a minimum value. As is well known, the Metropolis–Hastings algorithm [20] is prone to getting trapped in local minima, and takes many steps to escape these points, whereas our edge birth and death algorithm appears to move around the space more easily. In this particular realization, the Metropolis–Hastings algorithm takes about 1.25×10^6 steps to find AIC values as low as those found in only about 5×10^5 steps of the birth and death-type chain. That is, the birth and death chain is taking about 40% of the time of the Metropolis–Hastings algorithm to arrive at the same result.

6. CONCLUSIONS

We have presented a new algorithm for sampling from the posterior distribution of Bayesian networks given data based on a birth and death process. This new edge birth and death-type algorithm allows for probabilistic inferences to be made about Bayesian Network structures while avoiding some of the downfalls of existing methods. In particular, our edge birth and death process appears to mix well and does not get trapped in local extrema when trying to optimize a graph-scoring method.

An open question related to this new algorithm is whether it can be turned into a "perfect simulation" algorithm, removing all convergence error, using coupling-from-the-past. Such a construction would likely be similar to existing perfect algorithms (e.g. [2,3]) for spatial point processes.

https://doi.org/10.1017/S0269964817000432 Published online by Cambridge University Press

References

1. Akaike, H. (1981). Likelihood of a model and information criteria. *Journal of Econometrics* 16: 3–14.
2. Ambler, G.K. & Silverman, B.W. (2009). Perfect simulation of spatial point processes using dominated coupling from the past with application to a multiscale area-interaction point process. arXiv: 0903.2651.
3. Berthelsen, K.K. & Møller, J. (2002). A primer on perfect simulation for spatial point processes. *Bulletin Brazilian Mathematical Society* 33(3): 351–367.
4. Bockhorst, J., Craven, M., Page, D., Shavlik, J., & Glasner, J. (2003). A Bayesian network approach to operon prediction. *Bioinformatics* 19: 1227–1235.
5. Chen, T., He, H.L., & Church, G.M. (1999). Modeling gene expression with differential equations. In *Pacific Symposium Biocomputing* '99, pp. 29–40.
6. Cowell, R.G., Verall, R.J., & Yoon, Y.K. (2007). Modeling operational risk with Bayesian networks. *Journal of Risk and Insurance* 74: 795–827.
7. Curiac, D.I., Vasile, G., Banias, O., Volosencu, C., & Albu, A. (2009). Bayesian network model for diagnosis of psychiatric diseases. In *Proceedings of the ITI 2009 31st International Conference on Information Technology Interfaces*, pp. 61–66.
8. Dobra, A., Hans, C., Jones, B., Nevins, J.R., Yao, G., & West, M. (2004). Sparse graphical models for exploring gene expression data. *Journal of Multivariate Analysis* 90(1): 196–212.
9. Feller, W. (1968). *An introduction to probability theory and its applications. vol I.* London–New York–Sydney–Toronto: John Wiley & Sons.
10. Fenton, N. & Neil, M. (2013). *Risk Assessment and Decision Analysis with Bayesian Networks*. Boca Raton, FL: CRC Press, Taylor & Francis Group.
11. Friedman, N. & Koller, D. (2003). Being Bayesian about network structure. *Machine Learning* 50: 95–126.
12. Friedman, N., Linial, M., Nachman, I., & Pe'er, D. (2000). Using Bayesian networks to analyze expression data. *Journal of Computational Biology* 7: 601–620.
13. Griffiths, T.L., Kemp, C., & Tenenbaum, J.B. (2008). Bayesian models of cognition. *Cambridge Handbook of Computational Cognitive Modeling* (R. Son, Ed.). Cambridge, MA: Cambridge University Press, pp. 59–100.
14. Heckerman, D. (1995). A tutorial on learning with Bayesian networks. Technical Report, Microsoft Corporation.
15. Herskovits, E. (1991). *Computer-Based Probabilistic Network Construction*. Ph.D. dissertation, Stanford University, Medical Information Science.
16. Husmeier, D. (2003). Sensitivity and specificity of inferring genetic regulatory interactions from microarray experiments with dynamic Bayesian networks. *Bioinformatics* 19(17): 2271–2282.
17. Imoto, S., Kim, S., Goto, T., Miyano, S., Aburatani, S., Tashiro, K., & Kuhara, S. (2003). Bayesian network and nonparametric heteroscedastic regression for nonlinear modeling of genetic network. *Journal of Bioinformatics and Computational Biology* 1: 231–252.
18. Jarvis, E.D., Smith, V.A., Wada, K., Rivas, M.V., McElroy, M., Smulders, T.V., Carninci, P., Hayashizaki, Y., Dietrich, F., Wu, X., McConnell, P., Yu, J., Wang, P.P., Hartemink, A.J., & Lin, S. (2002). A framework for integrating the Songbird brain. *Journal of Comparative Physiology A* 188: 961–980.
19. Korb, K. & Nicholson, A. (2003). *Bayesian artificial intelligence*. Boca Raton, FL: Chapman and Hall/CRC.
20. Madigan, D. & York, J. (1995). Bayesian graphical models for discrete data. *International Statistical Review* 63: 215–232.
21. Mourad, R., Sinoquet, C., & Leray, P. (2011). A hierarchical bayesian network approach for linkage disequilibrium modeling and data-dimensionality reduction prior to genome-wide association studies. *BMC Bioinformatics* 12(1): 16.
22. Larrañaga, P., Calvo, B., Santana, R., Bielza, C., Galdiano, J., Inza, I., Lozano, J., Armañanzas, G.S.R., Pèrez, A., & Robles, V. (2005). Machine learning in bioinformatics. *Briefings in Bioinformatics* 7(1): 86–112.
23. Ong, I.M., Glasner, J.D., & Page, D. (2002). Modeling regulatory pathways in *E. coli* from time series expression profiles. *Bioinformatics* 18: 241–248.
24. Onisko, A., Druzdzel, M.J., & Wasyluk, H. (1999). A Bayesian network model for diagnosis of liver disorders. *Proceedings of the Eleventh Conference on Biocybernetics and Biomedical Engineering* 2: 842–846.
25. Preston, C. (1975). Spatial birth and death processes. *Advances in Applied Probability* 7(3): 465–466.
26. Ripley, B.D. (1977). Modelling spatial patterns. *Journal of the Royal Statistical Society. Series B*, 39(2): 172–212.

https://doi.org/10.1017/S0269964817000432 Published online by Cambridge University Press

27. Tenenbaum, J.B., Griffiths, T.L., & Kemp, C. (2006). Theory-based Bayesian models of inductive learning and reasoning. *Trends in Cognitive Sciences* 10(7): 309–318.
28. Weber, P., Medina-Oliva, G., Simon, C., & Iung, B. (2012). Overview on Bayesian networks applications for dependability, risk analysis and maintenance areas. *Engineering Applications of Artificial Intelligence*, 25(4): 671–682.
29. Xiang, Y., Pant, B., Eisen, A., Beddoes, M.P., & Pool, D. (1993). Multiply sectioned bayesian networks for neuromuscular diagnosis. *Artificial Intelligence in Medicine* 5: 293–314.

https://doi.org/10.1017/S0269964817000432 Published online by Cambridge University Press