Fast Bayesian Network Structure Learning using Quasi-Determinism Screening

Thibaud Rahier, Sylvain Marié, Stéphane Girard, Florence Forbes

Fast Bayesian Network Structure Learning using Quasi-Determinism Screening

Thibaud Rahier, Sylvain Marie, Stephane Girard, Florence Forbes
Univ. Grenoble Alpes, Inria, CNRS, G-INP and Schneider Electric

January 4, 2019

1 Introduction

Bayesian networks are probabilistic graphical models that present interest both in terms of knowledge discovery and density estimation. Learning Bayesian networks from data has been however proven to be NP-Hard by Chickering [1996].
There has been extensive work on tackling the ambitious problem of Bayesian network structure learning from observational data. In this paper, we focus on score-based structure learning: these algorithms rely on the definition of a network score, then on the search for the best-scoring structure among all possible directed acyclic graphs (DAGs).

Meanwhile, data itself may contain determinism, for example in the fields of cancer risk identification (de Morais et al. [2008]) or nuclear safety (Mabrouk et al. [2014]). Moreover, data is increasingly collected and generated by software systems whether in social networks, smart buildings, smart grid, smart cities or the internet of things (IoT) in general (Koo et al. [2016]). These systems in their vast majority rely on relational data models or lately on semantic data models (El Kaed et al. [2016]) which cause deterministic relationships between variables to be more and more common in datasets.

After reminding the background of Bayesian network structure learning (section 2), we propose the *quasi deterministic screening* algorithm (section 3). We then illustrate, using a benchmark dataset, that this algorithm is quicker and learns sparser structures than state of the art methods, for only a small decrease in performance score (section 4).
A more detailed version of this work, including theoretical results and proofs, as well as experiments on a wider range of datasets, is available in Rahier et al. [2018].

2 Bayesian network structure learning

2.1 Bayesian networks

Let $\mathbf{X} = (X_1, \ldots, X_n)$ be a n-tuple of categorical random variables with respective value sets $Val(X_1), \ldots, Val(X_n)$. The distribution of $\mathbf{X}$ is denoted by, $\forall\ \mathbf{x} = (x_1, \ldots, x_n) \in Val(\mathbf{X})$,

$$p(\mathbf{x}) = P(X_1 = x_1, \ldots, X_n = x_n).$$

For $I \subset [\![1, n]\!]$, we define $\mathbf{X}_I = \{X_i\}_{i \in I}$, and the notation $p(\cdot)$ and $p(\cdot|\cdot)$ is extended to the marginals and conditionals of any subset of variables: $\forall (\mathbf{x}_I, \mathbf{x}_J) \in Val(\mathbf{X}_{I \cup J})$, $p(\mathbf{x}_I|\mathbf{x}_J) = P(\mathbf{X}_I = \mathbf{x}_I | \mathbf{X}_J = \mathbf{x}_J)$.
Moreover, we suppose that D is a dataset containing M *i.i.d.* instances of $(X_1, \ldots, X_n)$. All quantities empirically computed from D will be written with a $.^D$ exponent (*e.g.* p^D refers to the empirical distribution with respect to D). Finally, D_I refers to the restriction of D to the obsevations of $\mathbf{X}_I$.

A Bayesian network is an object $\mathcal{B} = (G, \theta)$ where

- $G = (V, A)$ is a directed acyclic graph (DAG) structure with V the set of nodes and $A \subset V \times V$ the set of arcs. We suppose $V = [\![1, n]\!]$ where each node $i \in V$ is associated with the random variable X_i, and $\pi^G(i) = \{j \in V \ s.t.\ (j, i) \in A\}$ is the set of i's parents in G.
- $\theta = \{\theta_i\}_{i \in V}$ is a set of parameters. Each θ_i defines the local conditional distribution $P(X_i|\mathbf{X}_{\pi(i)})$.
  More precisely, $\theta_i = \{\theta_{x_i|\mathbf{x}_{\pi(i)}}\}$ where for $i \in V,\ x_i \in Val(X_i)$ and $\mathbf{x}_{\pi(i)} \in Val(\mathbf{X}_{\pi(i)})$,
  $$\theta_{x_i|\mathbf{x}_{\pi(i)}} = p(x_i|\mathbf{x}_{\pi(i)}).$$

A Bayesian network $\mathcal{B} = (G, \theta)$ encodes the following factorization of the distribution of $\mathbf{X}$: for $\mathbf{x} = (x_1, \ldots, x_n) \in Val(\mathbf{X})$,

$$p(\mathbf{x}) = \prod_{i=1}^{n} p(x_i|\mathbf{x}_{\pi^G(i)}) = \prod_{i=1}^{n} \theta_{x_i|\mathbf{x}_{\pi^G(i)}}.$$

Such a factorization notably implies that *each variable is independent of its non-descendents given its parents.*

2.2 Score-based approach to Bayesian network structure learning

Suppose we have a *scoring* function $s : DAG_V \to R$, where DAG_V is the set of all possible DAG structures with node set V. Score-based Bayesian network structure learning comes down to solving the following combinatorial optimization problem:

$$G^* \in \underset{G \in DAG_V}{\operatorname{argmax}}\, s(G). \tag{1}$$

It can be shown that $2^{\frac{n(n-1)}{2}} \leq |DAG_V| \leq 2^{n(n-1)}$ where $|V| = n$. There are therefore $2^{\mathcal{O}(n^2)}$ possible DAG structures containing n nodes: the size of DAG_V is said to be super-exponential in $|V|$.

The Max Log-Likelihood score
Most scoring functions used in practice are based on the likelihood function. The most straightforward being the Max log-likelihood (MLL) sore: for a given DAG structure $G \in DAG_V$, we define the MLL score of G wrt D as:

$$s^{MLL}(G : D) = \max_{\theta \in \Theta_G} l(\theta : D).$$

where $l(\theta : D) = \log(p_\theta(D))$ is the log-likelihood of θ given D and where Θ_G is the set of all θ's such that $\mathcal{B} = (G, \theta)$ is a Bayesian network.

The MLL score is straightforward and intuitive, but is maximized by complete DAGs. To solve this problem, one can either constrain the structure space (*e.g.* simply by restricting the maximum number of parents per nodes, or using more advanced methods as the MMHC algorithm introduced by Tsamardinos et al. [2006]), or use a score that expresses a goodness-of-fit vs complexity trade-off, such as BIC (Schwarz et al. [1978]) or BDe (Heckerman et al. [1995]).

3 Structure learning with quasi-determinism screening

3.1 Quasi-determinism

Our idea is to narrow the structure learning problem down to a subset of the original variables: the roots of a (quasi-)deterministic forest, in order to significantly decrease the overall computation time. This is what we call (quasi-)determinism screening.

Definition 1 *$\epsilon-$**quasi-determinism** ($\epsilon-$**qd**) Given a dataset D containing observations of variables X_i and X_j, the relationship $X_i \to X_j$ is $\epsilon-qd$ wrt D iff $H^D(X_j|X_i) \leq \epsilon$.*

where $H^D(X_i|X_j) = -\sum_{x_i, x_j} p^D(x_i, x_j) \log(p^D(x_i|x_j))$ is the empirical Shannon entropy. In the particular case where $\epsilon = 0$, we talk about *determinism* instead of *quasi-determinism*

We then define (quasi-)deterministic DAGs and forests as follows (the parameter ϵ is implicitly fixed in these two definitions).

Definition 2 ***(Quasi-)Deterministic DAG wrt** D*
$G \in DAG_V$ is said to be (quasi)-deterministic with respect to D iff $\forall i \in V$ s.t. $\pi^G(i) \neq \emptyset$, $\mathbf{X}_{\pi^G(i)} \to X_i$ is (quasi-)deterministic wrt D.

Definition 3 ***(Quasi-)Deterministic forest wrt*** D
$F \in DAG_V$ *is said to be a (quasi-)deterministic forest with respect to* D *iff* $F = \bigcup_{k=1}^{p} T_k$, *where* $T_1, \ldots, T_p$ *are* p *disjoint (quasi-)deterministic trees wrt* $D_{V_{T_1}}, \ldots, D_{V_{T_p}}$ *respectively and s.t.* $\bigcup_{k=1}^{p} V_{T_k} = V$.

where $\cup$ is the canonical union for graphs: $G \cup G' = (V_G \cup V_{G'}, A_G \cup A_{G'})$.

We show in Rahier et al. [2018] that a deterministic forest is the subgraph of an optimal DAG with respect to the MLL score, as stated in the following proposition. For a given forest F, we define the notation $R(F) = \{i \in V \mid \pi^F(i) = \emptyset\}$ to designate the set of F's roots (the union of the roots of each of its trees).

Proposition 1 *Suppose* F *is a deterministic forest wrt* D. *Let* $G^*_{R(F)}$ *be a solution of the structure learning optimization problem* (1) *for* $\mathbf{X}_{R(F)}$ *and the MLL score i.e.*

$$s^{MLL}(G^*_{R(F)} : D_{R(F)}) = \max_{G \in DAG_{R(F)}} s^{MLL}(G : D_{R(F)}).$$

Then, $G^* = F \cup G^*_{R(F)}$ *is a solution of* (1) *for* $\mathbf{X}$, *i.e.*

$$s^{MLL}(G^* : D) = \max_{G \in DAG_V} s^{MLL}(G : D).$$

Assumptions of Proposition 4 are always formally verified: if there is no determinism in the dataset D, then $R(F) = V$, and every tree T_k is formed of a single root node. In that case, solving problem (1) for $G^*_{R(F)}$ is the same as solving it for G^*. We are obviously interested in the case where $|R(F)| < n$, as this enables us to focus on a smaller structure learning problem while still having the guarantee to learn the optimal Bayesian network with regards to the MLL score.

Proposition 4, associated with the fact that forests are sparse DAGs, shows that deterministic forests are very promising with regards to the fit-complexity tradeoff (typically evaluated by scores such as BDe or BIC). Extending this intuition to $\epsilon-$quasi-determinism (presented in Definition 1), we now propose the *quasi-determinism screening* approach to Bayesian network structure learning.

3.2 Quasi-determinism screening algorithm

Algorithm 1 details how to find the simplest $\epsilon-$qd forest F_ϵ from a dataset D and a threshold ϵ. Here *simplest* refers to the complexity in terms of number of parameters.

This algorithm takes for input:

- D: a dataset containing M observations of $\mathbf{X}$,
- ϵ: a threshold for quasi-determinism.

Algorithm 1 Quasi-determinism screening (qds)

Input: D , ϵ

Compute empirical conditional entropy matrix $\mathbb{H}^D = \left(H^D(X_i|X_j)\right)_{1\leq i,j\leq n}$
for $i = 1$ to n **do** *#identify the set of potential $\epsilon-$qd parents for each i*
    compute $\pi_\epsilon(i) = \{j \in [\![1, n]\!] \setminus \{i\} \mid \mathbb{H}^D_{ij} \leq \epsilon\}$
for $i = 1$ to n **do** *#check for cycles in $\epsilon-$qd relations*
    if $\exists j \in \pi_\epsilon(i)$ *s.t.* $i \in \pi_\epsilon(j)$ **then**
        if $\mathbb{H}^D_{ij} \leq \mathbb{H}^D_{ji}$ **then**
            $\pi_\epsilon(j) \leftarrow \pi_\epsilon(j) \setminus \{i\}$
        else
            $\pi_\epsilon(i) \leftarrow \pi_\epsilon(i) \setminus \{j\}$
for $i = 1$ to n **do** *#choose the simplest among all potential parents*
    $\pi^*_\epsilon(i) \leftarrow \underset{j\in\pi_\epsilon(i)}{argmin} |Val(X_j)|$
Compute forest $F_\epsilon = (V_{F_\epsilon}, A_{F_\epsilon})$ where $V_{F_\epsilon} = [\![1, n]\!]$ and $A_{F_\epsilon} = \{(\pi^*_\epsilon(i), i) \mid i \in [\![1, n]\!]$ *s.t.* $\pi^*_\epsilon(i) \neq \emptyset\}$
Output: F_ϵ

3.3 Learning Bayesian networks using quasi-determinism screening

We now present Algorithm 2 (*qds-BNSL*), which uses quasi-determinism screening to accelerate Bayesian network structure learning. This algorithm takes as input:

- D: a dataset containing M observations of $\mathbf{X}$,
- ϵ: a threshold for quasi-determinism,
- *sota-BNSL*: a state of the art structure learning algorithm, taking for input a dataset, and returning a Bayesian network structure.

Algorithm 2 Bayesian network structure learning with quasi deterministic screening (qds-BNSL)

Input: D, ϵ, *sota-BNSL*

Compute F_ϵ by running **Algorithm 1** with input D and ϵ
Identify $R(F_\epsilon) = \{i \in [\![1, n]\!] \mid \pi^{F_\epsilon}(i) = \emptyset\}$, the set of F_ϵ's roots.
Compute $G^*_{R(F_\epsilon)}$ by running *sota-BNSL* on $D_{R(F_\epsilon)}$
$G^*_\epsilon \leftarrow F_\epsilon \cup G^*_{R(F_\epsilon)}$
Output: G^*_ϵ

3.4 Complexity analysis

Let *sota-BNSL* be a state of the art Bayesian network structure learning algorithm and $C_{sota}(M, n)$ be its complexity. The screening phase of Algorithm 2 implies $\mathcal{O}(Mn^2)$ operations. We can therefore write:

$$C_{Alg2}(M, n) = \mathcal{O}(Mn^2) + C_{sota}(M, n_r(\epsilon))$$

where $\forall \epsilon \geq 0,\ n_r(\epsilon) = |R(F_\epsilon)|$ is the number of roots of the forest F_ϵ.

$C_{sota}(M, n)$ is known to be typically exponential in n for the best exact structure learning algorithms, as those presented by Silander and Myllymäki [2006] or Bartlett and Cussens [2015], and it is expected to be significantly larger than $\mathcal{O}(Mn^2)$ for high-performing heuristics. We therefore expect an important decrease in computational time for Algorithm 2 compared to a state of the art algorithm, as long as $n_r(\epsilon)$ is sufficiently smaller than n.
We now present the experiments we conducted to confirm this intuition.

4 Experiments

4.1 Experimental setup

Data: In this paper we considered the `msnbc` dataset as preprocessed by Davis and Domingos [2010] and available on the UCI repository (Dheeru and Karra Taniskidou [2017]). It is the largest opensource categorical dataset with a small enough number of variables (17), allowing it to be displayed. Results on more than 15 other datasets are available in Rahier et al. [2018].
Programmation details and choice of *sota-BNSL*: After carefully evaluating several algorithms implemented in the `bnlearn R` package, we chose to use *Greedy Hill Climbing with* 10 *random restarts and a* $10-$*state long tabu list* as our state-of-the-art algorithm, as it consistently outperformed other built-in algorithms both in time and performance, in addition to being also used as a benchmark algorithm in the literature, notably by Teyssier and Koller [2005]. We now refer to this algorithm as *sota-BNSL*.
Evaluation of algorithms: We evaluated the algorithms using several quantitative measures on the learnt Bayesian networks: $BDeu$ score, $CVLL$ (Cross-Validated Log-Likelihood score, accounting for the graph's generalization performance), number of arcs and computation time.
Choice of ϵ for *qds-BNSL*: An approach for choosing ϵ in the case of the *qds-BNSL* algorithm is to pick values for $n_r(\epsilon)$, and manually find the corresponding values for ϵ: for a given dataset and $x \in [0, 1]$, we define $\epsilon_x = n_r^{-1}(\lfloor xn \rfloor)$, the value of ϵ for which the number of roots of the qd forest F_ϵ represents a proportion x of the total number of variables.

Figure 1, Figure 2 and Table 1 present the obtained Bayesian networks and associated evaluation criteria for *sota-BNSL* and *qds-BNSL* with $\epsilon = \epsilon_{0.5}$ (corresponding to a elimination by the screening phase of 50% of the original variables) on the `msnbc` dataset. Quantities displayed in Table 1 are the means of 20 runs with different seeds (all standard deviations are smaller than 0.05).

Choosing ϵ as explained previously is time-efficient but quite ad-hoc. In order to better grasp how the different evaluation criteria depend on ϵ, we display their evolution for different ϵ, ranging from 0 to $\max_i H^D(X_i) \approx 0.6$ in Figures 3 to 6.

In Figure 7, we compare the 'generalization performance vs number of arcs' tradeoff of the *qds-BNSL* algorithm to two other methods for learning sparse Bayesian networks: restricting the maximum number of parents allowed in *sota-BNSL* (range: 1 to 20), and decreasing the equivalent sample size (ESS) of the BDeu score used in *sota-BNSL*, inspired by Silander et al. [2007] (range: 10^{-10} to 5).

Figure 1: BN returned by *sota-BNSL*

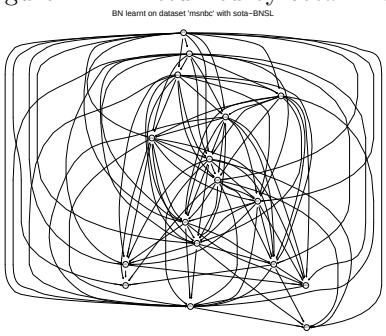
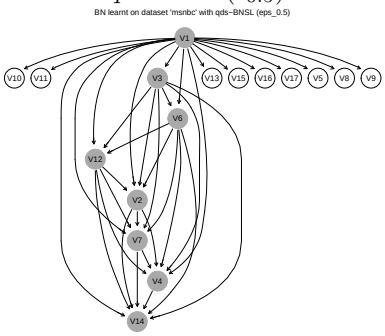


Figure 2: BN returned by *qds-BNSL*($\epsilon_{0.5}$)



Comparison of computation time, BDeu (normalized), CVLL (normalized), and number of arcs for the displayed Bayesian networks

	Figure 1 (**sota**)	Figure 2 ($\mathbf{qds}_{\epsilon_{0.5}}$)
t_{run} (sec)	252	36
BDe score	−6.2	−6.5
CVLL score	−6.1	−6.4
Nb arcs	102	37

Looking at Figure 1, Figure 2 and Table 1, we see how after the $\mathbf{qd}_{\epsilon_{0.5}}$-screening phase, half of the variables (corresponding to the nodes in white) are considered to be sufficiently explained by $V1$. They are therefore not taken into account by *sota-BNSL*, which is run only on the variables corresponding to the nodes in gray.

In the case of `msnbc`, this restriction of the learning problem implies only a small decrease in the final graph's performance (whether it is measured by the BDeu score or the CVLL score), while being 7 times faster to compute and enabling a significantly better readability (3 times as less arcs in the network).

As we can see on Figures 3 to 6, we could also use a smaller values of ϵ (*e.g.* smaller than 0.2) to obtain still interesting decreases in computation time with

Figure 3: CVLL score vs ϵ

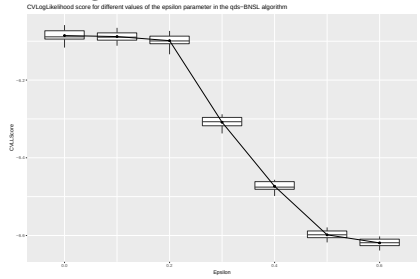


Figure 4: BDeu($4ESS = 5$) score vs ϵ

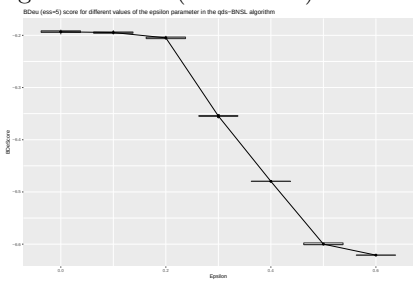


Figure 5: Computation time vs ϵ

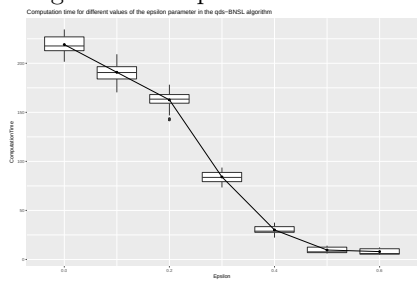
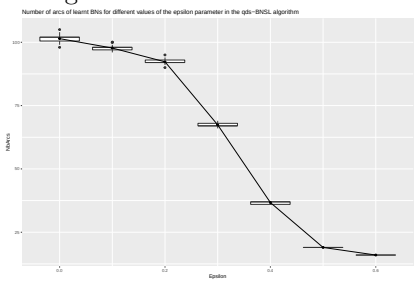


Figure 6: Number of arcs vs ϵ



close to no effect on the performance scores.

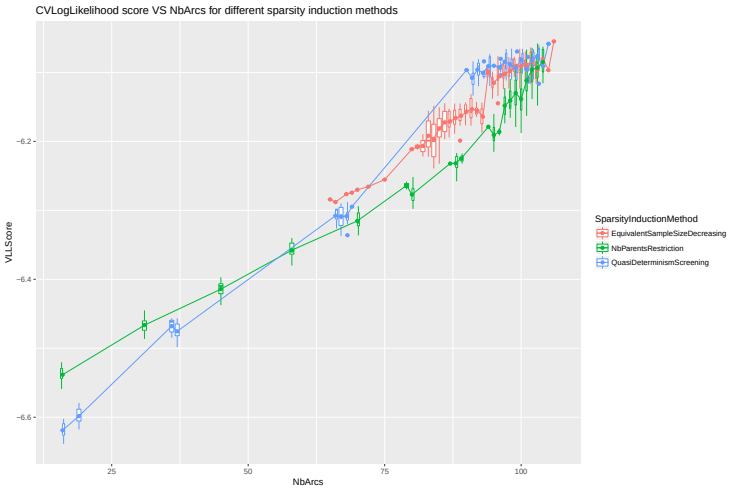
Results displayed in Figure 7 confirm that the *qds-BNSL* algorithm provides a very good tradeoff between readability (low number of arcs) and generalization performance (high CVLL score) as long as ϵ stays in a reasonable range (corresponding to the upper-right part of the plot).

5 Discussion

As it was expected from a theoretical point of view, the quasi-determinism screening approach to Bayesian network structure learning enables a significant decrease in computational time for a small decrease in graph scores. Moreover, this method provides an interesting tradeoff between readability (number of arcs) and generalization performance (CVLL) compared to other sparsity inducing structure learning algorithms, as long as ϵ is reasonably small compared to the variables' entropies.

The results are all the more promising that exact empirical determinism is barely present in the datasets used in this paper and its extended version (most of the time, $n_r(\epsilon = 0) = n$). We have also tested our algorithm on industrial descriptive metadatasets from the IoT domain, for which many variables possess (empirically) deterministic parents because of the underlying relational data schemas. In this context, *qds-BNSL* is up to 20 times faster than *sota-BNSL*, with often better learned graphs in terms of CVLL score. These results are dependent on very specific assumptions, which are however more and more often met by data accessible today, as previously noted.

Figure 7: CVLL score vs number of arcs for 3 approaches to learning sparse Bayesian networks



Our main research perspective is to find a principled way to choose ϵ without running the algorithm several times all the way through. This would save us important amounts of time, and prevent us from trying *qds-BNSL* on datasets that do not contain any strong pairwise relationships. Obtaining a bound such as the one presented in Proposition 4 of Rahier et al. [2018] seems like a promising way to achieve this goal. However, this bound concerns the MLL score and is far from tight in practice: we are currently searching for tighter bounds on the BDe or the BIC score of the graphs generated by *qds-BNSL*.

Finally, we could generalize our algorithm by making use of the fact that Proposition 4 holds when considering any deterministic DAGs (and not only forests), or by changing our definition of quasi determinism: one could choose the quantity $\frac{H^D(X|Y)}{H^D(X)}$ to describe the strength of the relationship $Y \to X$, which represents the proportion of X's entropy that is explained by Y. Moreover, $\frac{H^D(X|Y)}{H^D(X)} \leq \epsilon$ can be rewritten as $\frac{MI^D(X,Y)}{H(X)} \geq 1-\epsilon$, which gives another insight to quasi-determinism screening: for a given variable X, this comes down to finding a variable Y such that $MI^D(X,Y)$ is high. This is connected to the idea of Chow and Liu [1968], and later Cheng et al. [1997], for whom pairwise empirical mutual information is central.
In fact, it is quite straightforward to show that under the assumption of the existence of a deterministic tree T, both the algorithm by Chow and Liu [1968] and *qds-BNSL*($\epsilon = 0$) return trees that has the exact same MLL score as T. Further investigation of connections between these algorithms is part of our ongoing work.

References

Bartlett, M. and J. Cussens (2015). Integer Linear Programming for the Bayesian network structure learning problem. *Artificial Intelligence 1*, 1–14.

Cheng, J., D. A. Bell, and W. Liu (1997). Learning belief networks from data: An information theory based approach. In *Proceedings of the sixth international conference on Information and knowledge management*, pp. 325–331. ACM.

Chickering, D. M. (1996). Learning Bayesian networks is NP-complete. *Learning from data: Artificial intelligence and statistics V 112*, 121–130.

Chow, C. and C. Liu (1968). Approximating discrete probability distributions with dependence trees. *IEEE transactions on Information Theory 14*(3), 462–467.

Davis, J. and P. Domingos (2010). Bottom-up learning of Markov network structure. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 271–278.

de Morais, S. R., A. Aussem, and M. Corbex (2008). Handling almost-deterministic relationships in constraint-based Bayesian network discovery: Application to cancer risk factor identification. In *European Symposium on Artificial Neural Networks, ESANN'08*.

Dheeru, D. and E. Karra Taniskidou (2017). UCI machine learning repository.

El Kaed, C., B. Leida, and T. Gray (2016). Building management insights driven by a multi-system semantic representation approach. In *Internet of Things (WF-IoT), 2016 IEEE 3rd World Forum on*, pp. 520–525. IEEE.

Heckerman, D., D. Geiger, and D. M. Chickering (1995). Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning 20*(3), 197–243.

Koo, D. D., J. J. Lee, A. Sebastiani, and J. Kim (2016). An internet-of-things (iot) system development and implementation for bathroom safety enhancement. *Procedia Engineering 145*, 396–403.

Mabrouk, A., C. Gonzales, K. Jabet-Chevalier, and E. Chojnacki (2014). An efficient Bayesian network structure learning algorithm in the presence of deterministic relations. In *Proceedings of the Twenty-first European Conference on Artificial Intelligence*, pp. 567–572. IOS Press.

Rahier, T., S. Marié, S. Girard, and F. Forbes (2018, March). Fast Bayesian Network Structure Learning using Quasi-Determinism Screening. working paper or preprint.

Schwarz, G. et al. (1978). Estimating the dimension of a model. *The Annals of Statistics 6*(2), 461–464.

Silander, T., P. Kontkanen, and P. Myllymäki (2007). On Sensitivity of the MAP Bayesian Network Structure to the Equivalent Sample Size Parameter. *Proceedings of the Twenty-Third Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-07)*, 360–367.

Silander, T. and P. Myllymäki (2006). A simple approach for finding the globally optimal Bayesian network structure. *Networks*, 445–452.

Teyssier, M. and D. Koller (2005). Ordering-based search: A simple and effective algorithm for learning Bayesian networks. In *Proceedings of the Twenty-first Conference on Uncertainty in AI (UAI)*, pp. 584–590.

Tsamardinos, I., L. E. Brown, and C. F. Aliferis (2006). The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning 65*(1), 31–78.