A STRUCTURAL RUNTIME LAW FOR SCALABLE EXACT INFERENCE IN TWO-LAYER NOISY-OR BAYESIAN NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Exact inference in two-layer noisy-OR Bayesian networks (BN2Os) is challenging due to the multiply connected structure. We identify a structural factorization of BN2O joint distributions induced by the unconditional independence of root variables. To exploit this factorization, we introduce noisy-OR circuit engineering (NOICE), a circuit framework that compiles BN2Os into a quotient of family-set marginal distributions represented by explicit arithmetic subcircuits. This factorization yields a structural runtime law for exact inference, $t(Q) = c(N)2^{\omega(Q)}$, where Q is a query, $c(N)$ is the cost of a single NOICE circuit propagation, and $\omega(Q)$ is the number of non-queried roots shared by two or more observed leaves. We prove that NOICE compilation produces circuits whose size grows linearly in the numbers of roots, leaves, and edges, with closed-form expressions determining the compiled size *a priori*. Experiments on BN2Os spanning five orders of magnitude validate the predicted runtime law for inference and demonstrate compilation of BN2Os with up to 24M edges, extending the scale of exact structural compilation by up to five orders of magnitude over prior general-purpose methods.

1 INTRODUCTION

Two-layer noisy-OR Bayesian networks (BN2Os) (Heckerman, 1990; Pearl, 1988) provide a canonical model for reasoning with many independent causes and effects and are widely studied in applications such as medical diagnosis and causal machine learning (Dedieu et al., 2023; Richens et al., 2020; Auzina & Tomczak, 2021). Recent work on probabilistic circuits emphasizes how structural constraints govern expressivity, tractability, and reuse (Choi et al., 2020; Vergari et al., 2021; Wang & Kwiatkowska, 2022). This perspective has motivated analyses of tree-structured probabilistic circuits (Yin & Zhao, 2024), tractability conditions for structured decomposable probabilistic circuits (Wang & Kwiatkowska, 2023; 2022), circuit restructuring (Zhang et al., 2025), and Bayesian structure scoring for deterministic circuit models (Yang et al., 2023). These developments highlight the potential of structured circuit representations and motivate studying BN2Os as a principled testbed for structural circuit design.

Exact inference in dense two-layer Bayesian networks has long been regarded as computationally infeasible (Ng & Jordan, 1999). Although BN2Os admit a compact parameterization, their multiply connected structure makes exact reasoning challenging due to dense interactions among shared parents. Prior approaches include arithmetic circuit methods based on the framework of Darwiche (2003), such as the Arithmetic Circuit Evaluator (ACE) (UCLA Automated Reasoning Group, 2015) and the rank-one tensor construction method (Vomlel & Savický, 2008), as well as search and model-counting methods (Gogate & Domingos, 2010; Li et al., 2008; 2011). Within the circuit-based paradigm, ACE employs parent divorcing (Olesen et al., 1989) to exploit the functional decomposition of noisy-OR conditional probability tables, followed by triangulation for compilation, whereas the rank-one tensor construction factorizes each noisy-OR conditional probability table into two-dimensional factors and transforms the BN2O into an undirected model triangulated for compilation (Vomlel & Savický, 2008). Consequently, existing circuit-based approaches exploit the noisy-OR structure of BN2Os but rely on triangulation-based representations that do not preserve the mutual independence of root variables in a form that can be exploited during inference. Early work on BN2Os noted that general-purpose Bayesian network algorithms do not exploit the para-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

metric structure of noisy-OR conditional probability tables and suggested that useful factorizations of the noisy-OR parameterization could enable more efficient reasoning (Jaakkola & Jordan, 1999).

In this work, we identify such a factorization for BN2Os and show that it enables scalable circuit compilation and exact inference. Our approach leverages the unconditional independence of BN2O root nodes during circuit compilation. Because the roots are mutually independent, each parent-set marginal factorizes into a product of singleton root marginals. This independence induces an exact family-set factorization of the BN2O joint distribution. Remarkably, this factorization is identical in form to that of singly connected Bayesian networks (polytrees), despite the multiply connected topology of BN2Os. It therefore motivates a structural compilation strategy that directly encodes this factorization rather than moralizing the network.

We formalize this strategy as noisy-OR circuit engineering (NOICE), a structural circuit representation for exact inference in BN2Os. NOICE compiles each family-set marginal as a tractable arithmetic subcircuit and composes these according to the induced factorization. Unlike classical arithmetic circuit approaches that encode the full combinatorics of the BN2O inside triangulation-based compiled representations, NOICE maintains a circuit representation that directly preserves the underlying BN2O independence structure. We prove that the resulting circuit size is linear in the numbers of roots, leaves, and edges of the BN2O. While worst-case inference complexity remains $O(2^r)$ for r root variables, practical runtime depends on the number of non-queried shared roots in a query. This yields the BN2O Overlap Law for exact inference in BN2Os using NOICE circuits, $t(Q) = c(N)2^{\omega(Q)}$, where Q is a query, $c(N)$ denotes the cost of a single propagation in a NOICE circuit with N edges, and $\omega(Q)$ counts non-queried roots shared by two or more observed leaves.

Experiments demonstrate the practical impact of NOICE across five orders of magnitude in BN2O size. Whereas existing general-purpose compilation methods scale up to roughly 240 edges on BN2Os with four parents per leaf, NOICE compiles BN2Os with up to 24M edges, extending the scale of exact structural compilation by up to five orders of magnitude. Moreover, runtime grows exponentially with the number of non-queried roots shared by two or more observed leaves while remaining linear in the number of circuit runs, yielding a predictable structural cost model with low relative prediction error. All proofs omitted from the main text appear in Appendix A.

2 PRELIMINARIES

We review the structure of two-layer noisy-OR Bayesian networks and establish our notation. In a BN2O, Boolean variables are partitioned into roots $\boldsymbol{X} = \{X_1, \ldots, X_r\}$ and leaves $\boldsymbol{Y} = \{Y_1, \ldots, Y_s\}$. Upper-case letters (e.g., X, Y, Z) denote variables, and lower-case letters denote their values. For a Boolean variable X, we write x to denote $X = 1$ (true) and $\bar{x}$ to denote $X = 0$ (false). Sets of variables are denoted by bold-face upper-case letters (e.g., $\boldsymbol{Z}$), and their instantiations by bold-face lower-case letters (e.g., $\boldsymbol{z}$), where $\mathbf{0}$ denotes the all-zero configuration. The domain of a set of variables $\boldsymbol{Z}$ is denoted by $\mathrm{dom}(\boldsymbol{Z})$, i.e., the set of all joint assignments to $\boldsymbol{Z}$.

Given a node Z in a directed acyclic graph, its parents (immediate predecessors) are denoted by π_Z, and its children (immediate descendants) by $ch(Z)$; for a set of variables $\boldsymbol{Z}$, $\pi_{\boldsymbol{Z}} = \bigcup_{Z \in \boldsymbol{Z}} \pi_Z$. For roots X_i, let $d_i = |ch(X_i)|$. For a leaf variable Y_j, we define its family as $F_j = \{Y_j\} \cup \pi_j$, where π_j abbreviates π_{Y_j}. A Bayesian network (Pearl, 1988) associates each node Z with a conditional probability table $P(Z \mid \pi_Z)$, yielding the joint distribution $P(\boldsymbol{Z}) = \prod_{Z \in \boldsymbol{Z}} P(Z \mid \pi_Z)$.

Definition 1 (BN2O (Heckerman, 1990; Pearl, 1988))**.** *A two-layer noisy-OR Bayesian network (BN2O) $\mathcal{B}$ is a bipartite directed acyclic graph whose nodes are partitioned into root nodes $\boldsymbol{X} = \{X_1, \ldots, X_r\}$ and leaf nodes $\boldsymbol{Y} = \{Y_1, \ldots, Y_s\}$. Root nodes have no parents and are therefore mutually independent a priori. We assume each root has at least one child. For each leaf Y_j, directed edges $X_i \to Y_j$ define the parent set $\pi_{Y_j} = \{X_i \in \boldsymbol{X} \mid X_i \to Y_j\}$. Each edge is associated with an inhibitor probability $q_i^j \in [0, 1]$, representing the probability that an active root X_i fails to activate leaf Y_j.*

In a non-leaky model, the probability that leaf Y_j is absent given parent configuration $\boldsymbol{x}$ is

$$P(\bar{y}_j \mid \pi_{Y_j} = \boldsymbol{x}) = \prod_{i : X_i \in \pi_{Y_j},\, x_i = 1} q_i^j, \tag{1}$$

where $P(y_j \mid \pi_{Y_j} = \boldsymbol{x}) = 1 - P(\bar{y}_j \mid \pi_{Y_j} = \boldsymbol{x})$ and $P(y_j \mid \pi_{Y_j} = \mathbf{0}) = 0$.

The unconditional independence of roots underlies the BN2O structure and our compilation strategy.

Remark. *Following Heckerman (1990), we adopt the standard multiplicative noisy-OR parameterization and present the non-leaky form for clarity. In the leaky formulation, each leaf Y_j has its own leak parameter, which can be represented as an additional parent of that leaf. All structural and complexity results extend unchanged to this formulation.*

Each leaf Y may be observed as a positive finding (y), a negative finding ($\bar{y}$), or remain unobserved. A typical BN2O inference task is to compute posterior probabilities such as $P(X = x \mid \boldsymbol{Y} = \boldsymbol{y})$. This can be expressed as computing marginals $P(\boldsymbol{Z} = \boldsymbol{z})$, whose evaluation involves only the conditional probability tables of variables in $\boldsymbol{Z} \cup \pi_{\boldsymbol{Z}}$. Inference complexity in classical approaches is driven by the number of positive findings that induce overlapping active parents (Heckerman, 1990); in NOICE, this dependence is captured more generally by the number of non-queried roots shared by two or more observed leaves.

3 Structural factorization and NOICE circuits

Although BN2Os are multiply connected, we show that their joint distribution admits an exact family-set factorization identical in form to that of polytrees. This structural result arises from the unconditional independence of the root variables and enables a direct compilation strategy that preserves this independence, leading to the NOICE circuit representation.

Structural factorization of BN2Os. We begin by deriving the structural factorization underlying NOICE circuits. Let $\mathcal{B}$ be a BN2O with roots $\boldsymbol{X}$ and leaves $\boldsymbol{Y}$. For root nodes $X_i \in \boldsymbol{X}$, $\pi_{X_i} = \emptyset$, we adopt the convention $P(\emptyset) = 1$, and assume strictly positive root priors. A Bayesian network factorization can be rewritten in terms of marginals $P(\boldsymbol{X}, \boldsymbol{Y}) = \prod_{Z \in \boldsymbol{X} \cup \boldsymbol{Y}} \frac{P(F_Z)}{P(\pi_Z)}$.

Theorem 1 (Family-Set Factorization)**.** *Let $\mathcal{B}$ be a BN2O with roots $\boldsymbol{X}$ and leaves $\boldsymbol{Y}$. Let $d_i = |ch(X_i)|$ be the number of children of X_i. Then*

$$P(\boldsymbol{X}, \boldsymbol{Y}) = \frac{\prod_{Y_j \in \boldsymbol{Y}} P(F_j)}{\prod_{X_i \in \boldsymbol{X}} P(X_i)^{d_i - 1}}. \tag{2}$$

Despite being multiply connected and restricted to two levels, BN2Os admit a joint distribution factorization identical in form to that of polytrees, exposing reusable marginal components that can be compiled independently. In a polytree, the joint distribution can be written as $P(\boldsymbol{Z}) = \prod_{Z \in \boldsymbol{Z}} \frac{P(F_Z)}{P(Z)^{|ch(Z)|}}$ (Darwiche, 2009). In a BN2O, the bipartite structure ensures that roots have no parents and leaves have no children, so only root nodes contribute nonzero multiplicities in the denominator. For each root X_i, the marginal $P(X_i)$ appears once in the numerator through its own family and d_i times through the parent sets of its children. Cancelling the numerator occurrence yields exponent $d_i - 1$ for each root X_i, recovering (2). NOICE circuits directly encode this factorization, yielding a constructive compilation of BN2Os that preserves root independence while making the multiplicity structure explicit.

Compiling NOICE circuits. The constructive form of Theorem 1 suggests a compilation strategy: represent each family-set marginal as a subcircuit and combine them using multiplication and quotient operations according to the factorization, enabling independent compilation and reuse.

Definition 2 (NOICE Circuit)**.** *Let $\mathcal{B}$ be a BN2O with roots $\boldsymbol{X}$ and leaves $\boldsymbol{Y}$. A* NOICE circuit *$\mathcal{C}$ for $\mathcal{B}$ is a rooted directed acyclic graph with root division node* Div *having exactly two children,* Num *and* Den*. The multiplication node* Num *has children that are subcircuits $\mathcal{C}(F_j)$ for each $Y_j \in \boldsymbol{Y}$, and the multiplication node* Den *has children consisting of $d_i - 1$ copies of each subcircuit $\mathcal{C}(X_i)$ for $X_i \in \boldsymbol{X}$, where $d_i = |ch(X_i)|$. The circuit $\mathcal{C}$ evaluates to the factorization in Eq.* (2)*, where each $\mathcal{C}(F_j)$ and $\mathcal{C}(X_i)$ is a subcircuit computing the corresponding marginal.*

NOICE circuits employ the quotient operation (Vergari et al., 2021), allowing a division node in addition to summation and multiplication nodes.

Each family-set marginal is implemented using a fixed circuit template. Figure 1(a) shows the subcircuit for the root X_i (family set $F_i = \{X_i\}$). Here, θ_{x_i} and $\theta_{\bar{x}_i}$ denote the prior probabilities $P(x_i)$ and $P(\bar{x}_i)$, respectively, and indicator nodes λ_{x_i} and $\lambda_{\bar{x}_i}$ are described later. Figure 1(b)

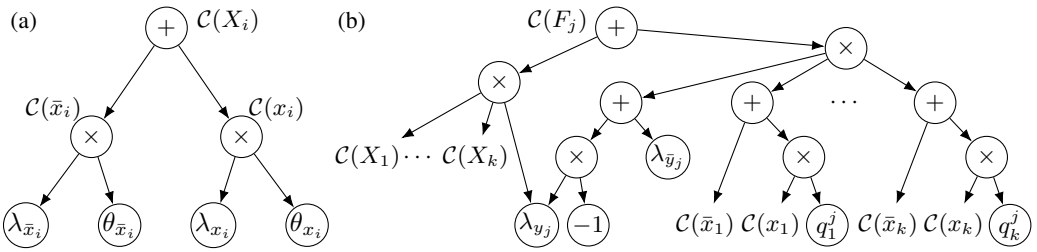


Figure 1: Subcircuits (a) $\mathcal{C}(X_i)$ for root X_i and (b) $\mathcal{C}(F_j)$ for family $F_j = \{Y_j\} \cup \pi_j$ with $k = |\pi_j|$ parents. Here $\mathcal{C}(X_i)$ encodes $P(X_i)$, while $\mathcal{C}(F_j)$ encodes $P(F_j)$.

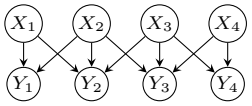
Algorithm 1 NOICE circuit compilation

Input: BN2O $\mathcal{B}$
Output: NOICE circuit $\mathcal{C}$
Build subcircuit $\mathcal{C}(X_i)$ for each root $X_i \in \boldsymbol{X}$ ▷ $\mathcal{C}(X_i)$ template in Figure 1(a)
Build subcircuit $\mathcal{C}(F_j)$ for each leaf $Y_j \in \boldsymbol{Y}$ ▷ $\mathcal{C}(F_j)$ template in Figure 1(b)
Create the multiplication node Num with children $\{\, \mathcal{C}(F_j) \mid Y_j \in \boldsymbol{Y} \,\}$ ▷ construct numerator
Create the multiplication node Den ▷ construct denominator
for each root $X_i \in \boldsymbol{X}$ **do**
    for $l = 1, \ldots, d_i - 1$ **do**
        Add an edge from Den to $\mathcal{C}(X_i)$ ▷ reuses $\mathcal{C}(X_i)$ $d_i - 1$ times
    end for
end for
Create the root division node Div with children Num and Den ▷ computes Num/Den
return $\mathcal{C}$

shows the template for a leaf family $F_j = \{Y_j\} \cup \pi_j$ with $k = |\pi_j|$ parents, including indicator nodes λ_{y_j} and $\lambda_{\bar{y}_j}$. The structure of the leaf family-set subcircuit corresponds to a linear arithmetic representation of a noisy-OR gate noted by Vomlel & Savický (2008), which we interpret as a circuit for the family-set marginal $P(F_j) = P(Y_j, \pi_j)$. The leaf template introduces a constant number of nodes per parent and therefore scales linearly in k (Vomlel & Savický, 2008). These templates uniquely determine the node types and edge structure introduced during compilation.

We construct NOICE circuits directly from the factorization in Eq. (2). All leaf family-set marginals are realized by subcircuits $\mathcal{C}(F_j)$ and combined in the multiplication node Num, yielding $\mathsf{Num} = \prod_{Y_j \in \boldsymbol{Y}} P(F_j)$. The denominator node Den realizes $\prod_{X_i \in \boldsymbol{X}} P(X_i)^{d_i - 1}$ by connecting each root-marginal subcircuit $\mathcal{C}(X_i)$ with multiplicity $d_i - 1$, reusing the same subcircuit as required. The root division node Div computes Num/Den, yielding a circuit that evaluates exactly to this factorization. NOICE circuits are formally constructed by Algorithm 1.

Example 1. *Consider the following BN2O, with families $F_j = \{Y_j\} \cup \pi_j$. By Theorem 1, the joint distribution $P(\boldsymbol{X}, \boldsymbol{Y})$ factorizes as*



$$P(\boldsymbol{X}, \boldsymbol{Y}) = \frac{P(F_1)P(F_2)P(F_3)P(F_4)}{P(X_1)P(X_2)^2P(X_3)^2P(X_4)}.$$

Algorithm 1 instantiates subcircuits for $P(X_1), \ldots, P(X_4)$ and $P(F_1), \ldots, P(F_4)$, forms a product node Num *with children $\mathcal{C}(F_1), \ldots, \mathcal{C}(F_4)$, and forms* Den *as a product node with children $\mathcal{C}(X_1), \ldots, \mathcal{C}(X_4)$, where $\mathcal{C}(X_i)$ appears $d_i - 1$ times (i.e., $d_1 - 1 = 1$, $d_2 - 1 = 2$, $d_3 - 1 = 2$, $d_4 - 1 = 1$). The root division node* Div *with two children* Num *and* Den *computes* Num/Den*, yielding the compiled NOICE circuit, matching the BN2O factorization above.*

The structural regularity of NOICE compilation yields an exact characterization of circuit size.

Compilation size analysis. We measure circuit size by its number of edges, following the standard convention in the arithmetic circuit literature. The structural regularity of NOICE circuits yields an exact closed-form characterization of compilation size. In particular, the number of nodes and edges in the NOICE circuit depends only on the numbers of roots, leaves, and edges in the BN2O.

Algorithm 2 NOICE circuit evaluation (NCE)

1: **Input:** NOICE circuit $\mathcal{C}$ compiled from BN2O $\mathcal{B}$, query $P(\boldsymbol{Z} = \boldsymbol{z})$
2: **Output:** Exact value $P(\boldsymbol{Z} = \boldsymbol{z})$
3: Initialize all indicators to 1
4: $\boldsymbol{S} \leftarrow \{ X_i \in \boldsymbol{X} \setminus \boldsymbol{Z} \mid |ch(X_i) \cap \boldsymbol{Z}| \geq 2 \}$ ▷ shared roots
5: $\eta \leftarrow 0$ ▷ initialize accumulator
6: **if** $\boldsymbol{S} = \emptyset$ **then**
7: $\quad \eta \leftarrow$ Propagation on $\mathcal{C}$ with indicators set for $\boldsymbol{Z} = \boldsymbol{z}$ ▷ no overlap: one pass
8: **else**
9: $\quad$ **for** each configuration $\boldsymbol{s} \in \mathrm{dom}(\boldsymbol{S})$ **do** ▷ sum across overlap assignments
10: $\quad\quad \eta \leftarrow \eta +$ Propagation on $\mathcal{C}$ with indicators set for $\boldsymbol{Z} = \boldsymbol{z}$ and $\boldsymbol{S} = \boldsymbol{s}$
11: $\quad$ **end for**
12: **end if**
13: **return** η

Theorem 2 (Linear Compilation Size)**.** *Let $\mathcal{B}$ be a BN2O with r roots, s leaves, and t directed edges, and let $\mathcal{C}$ be the NOICE circuit produced by Algorithm 1. Then $\mathcal{C}$ has exactly $7r+8s+3t+3$ nodes and $5r + 9s + 7t + 2$ edges. In particular, the number of circuit edges is linear in $r + s + t$.*

Proof. Figure 1(a) shows that each root subcircuit $\mathcal{C}(X_i)$ has 7 nodes and 6 edges, giving $7r$ nodes and $6r$ edges over all r roots. In Figure 1(b), the sub-DAG in $\mathcal{C}(F_j)$ induced by $\{\lambda_{y_j}, \lambda_{\bar{y}_j}, -1\}$ and their ancestors contributes 8 nodes and 8 edges per leaf, requiring $8s$ nodes and $8s$ edges. Each of the t BN2O edges adds 3 nodes and 6 edges within its family-set subcircuit, for an additional $3t$ nodes and $6t$ edges. The numerator node Num adds 1 node and s edges, while the denominator node Den multiplies $t - r$ root marginals, since $\sum_{i=1}^{r}(d_i - 1) = t - r$, adding 1 node and $t - r$ edges. Finally, one division node Div and 2 edges combine Num and Den. Summing yields $7r+8s+3t+3$ nodes and $5r + 9s + 7t + 2$ edges, both linear in $r + s + t$. □

The counts r, s, and t therefore determine the compiled circuit size *a priori*, without compilation.

Example 2. *The BN2O in Example 1 has $r = 4$ roots, $s = 4$ leaves, and $t = 10$ edges. The resulting NOICE circuit therefore has exactly 93 nodes and 128 edges, determined directly from r, s, and t.*

Exact inference. Exact inference in a general Bayesian network with n variables has worst-case complexity $O(n\,\delta^{\tau})$, where τ denotes the treewidth of the moralized graph and δ is the maximum variable cardinality. BN2Os form a structurally restricted subclass of multiply connected networks in which dependencies arise through shared parents of leaf variables; consequently, inference complexity can be characterized by the extent to which observed leaves share common roots.

We describe exact inference in a NOICE circuit. Indicator handling and circuit propagation follow arithmetic circuits for Bayesian networks (Darwiche, 2003), except that the compiled structure is specialized to BN2Os. Each variable is associated with two indicator nodes. For a leaf Y_j, indicators λ_{y_j} and $\lambda_{\bar{y}_j}$ are set according to the observation (e.g., $\lambda_{y_j} = 1, \lambda_{\bar{y}_j} = 0$ for a positive finding $Y_j = y_j$; $\lambda_{y_j} = 0, \lambda_{\bar{y}_j} = 1$ for a negative finding $Y_j = \bar{y}_j$; and $\lambda_{y_j} = \lambda_{\bar{y}_j} = 1$ if Y_j is unobserved). For a root X_i, indicators λ_{x_i} and $\lambda_{\bar{x}_i}$ encode the assignment during evaluation. In an arithmetic circuit, answering a probability-of-evidence query $P(\boldsymbol{Z} = \boldsymbol{z})$ requires a single circuit evaluation with indicators fixed to encode $\boldsymbol{Z} = \boldsymbol{z}$. In contrast, in a NOICE circuit, indicators for observed leaves remain fixed across evaluations, but one propagation is performed for each assignment to the shared roots $\boldsymbol{S}$ induced by the observations, as formalized in Algorithm 2. Thus, multiple circuit evaluations may be required, differing only in the indicator settings of overlapping roots. Positive findings ($Y_j = y_j$) and negative findings ($Y_j = \bar{y}_j$) incur identical cost in a NOICE circuit.

Example 3. *We compute $P(\boldsymbol{Z} = \boldsymbol{z})$ with $\boldsymbol{Z} = \{Y_1, Y_3\}$ and $\boldsymbol{z} = (y_1, y_3)$ using Algorithm 2 in the NOICE circuit from Example 1. Line 4 yields $\boldsymbol{S} = \{X_2\}$, the only root whose children intersect both observed leaves in $\boldsymbol{Z}$. The algorithm evaluates configurations $x_2, \bar{x}_2 \in \mathrm{dom}(\boldsymbol{S})$ while holding $Y_1 = y_1$ and $Y_3 = y_3$ fixed, summing resulting circuit evaluations, with all other indicators set to 1:*

$$P(Y_1 = y_1, Y_3 = y_3) = \sum_{X_2} \frac{P(Y_1 = y_1, X_2)\, P(X_2, Y_3 = y_3)}{P(X_2)}. \tag{3}$$

Example 3 shows that one circuit propagation is required per assignment to $\boldsymbol{S}$.

We now establish the correctness of Algorithm 2.

Theorem 3 (Soundness)**.** *Let $\mathcal{C}$ be the NOICE circuit produced by Algorithm 1 from a BN2O $\mathcal{B}$. For any complete instantiation $(\boldsymbol{x}, \boldsymbol{y})$, evaluating $\mathcal{C}$ under the corresponding indicator setting yields $P(\boldsymbol{x}, \boldsymbol{y})$. Consequently, for any query $P(\boldsymbol{Z} = \boldsymbol{z})$ with $\boldsymbol{Z} \subseteq \boldsymbol{X} \cup \boldsymbol{Y}$, Algorithm 2 returns the exact value $P(\boldsymbol{Z} = \boldsymbol{z})$.*

The computational complexity of exact inference in NOICE circuits is analyzed next.

Theorem 4 (Inference Complexity)**.** *Let $\mathcal{B}$ be a BN2O with r root nodes, and let $\boldsymbol{S}$ be the set of non-queried roots shared by two or more observed leaves for a query $P(\boldsymbol{Z} = \boldsymbol{z})$. Then Algorithm 2 performs exactly $2^{|\boldsymbol{S}|}$ circuit propagations. In particular, inference requires $O(2^{|\boldsymbol{S}|})$ propagations, and since $|\boldsymbol{S}| \leq r$, worst-case complexity is $O(2^r)$ circuit propagations.*

For a query Q, $|\boldsymbol{S}|$ depends only on the number of non-queried roots shared by at least two observed leaves and is independent of the total number of leaves. Accordingly, inference complexity in BN2Os depends on overlap among non-queried roots rather than network size.

In the fully connected BN2O case, every root is a parent of every leaf, and moralization induces a clique of size $r + 1$ over each leaf and its r parents, implying the treewidth is at least r. Since the moralized graph is triangulated, the treewidth is r. As queries exist with $|\boldsymbol{S}| = r$, $\Theta(2^r)$ circuit propagations are required. Thus, the $O(2^r)$ bound is tight for fully connected BN2Os.

While worst-case inference requires $\Theta(2^r)$ circuit propagations, the compiled circuit size is $O(r + s + t)$. Thus NOICE achieves linear representation size while incurring query-time summation over the roots in $\boldsymbol{S}$ in the worst case. Each assignment to $\boldsymbol{S}$ requires exactly one circuit propagation, so a query Q requires $2^{|\boldsymbol{S}|}$ propagations in total. If $c(N)$ denotes the time required for a single propagation in a NOICE circuit of size N, the runtime of a query can therefore be written as

$$t(Q) = c(N)\, 2^{\omega(Q)}, \tag{4}$$

where $\omega(Q) = |\boldsymbol{S}|$ is the number of non-queried roots shared by multiple observed leaves. We refer to this relation as the BN2O Overlap Law, which states that inference cost in NOICE circuits grows exponentially with the overlap index $\omega(Q)$. Section 4 empirically validates this structural prediction.

4 Experimental results

Overview. We empirically validate the compilation and inference complexity predictions of NOICE circuits. We evaluate (i) whether circuit size scales linearly and compilation time scales superlinearly with BN2O size, enabling compilation of million-edge networks; (ii) whether exact inference cost is governed by the query-induced overlap index $\omega(Q)$ rather than evidence size; and (iii) whether runtime follows the BN2O Overlap Law $t(Q) = c(N)2^{\omega(Q)}$, yielding a predictive runtime model based on $\omega(Q)$. Across all networks, circuit size scales linearly and compilation time superlinearly with BN2O size, inference cost depends on root overlap, and runtime follows this law.

Setup. Experiments were conducted on Ubuntu 24.04.3 LTS using a 3.3 GHz AMD Ryzen 5 5600H (6 cores/12 threads) with 64 GB (59.6 GiB) RAM; all experiments were performed on a CPU with a one-hour timeout per query. We evaluate two BN2O families. The first consists of synthetic networks with fixed fan-in of four disease parents per leaf, a standard benchmark in prior BN2O studies; each leaf also includes a leak parameter, which is not counted among the four disease parents or in reported BN2O edge totals, but is included in compiled circuit size counts. Network sizes span five orders of magnitude, ranging from 40 to 24M disease–finding edges, corresponding to (10–10–40) to (6M–6M–24M) in roots–leaves–edges. The second family consists of QMR–DT–like networks (Shwe et al., 1991) generated using a truncated power-law leaf fan-in with preferential attachment to induce hub parents; each leaf again includes a leak parameter. Network sizes range from 180 to 13.5M disease–finding edges. To evaluate compilation scalability, we compare NOICE against methods that construct a reusable inference representation after a single compilation, specifically ACE (UCLA Automated Reasoning Group, 2015), a widely used arithmetic circuit compiler that exploits noisy-OR semantics and remains a leading baseline (Agrawal et al., 2021), and the rank-one tensor construction (Vomlel & Savický, 2008). Recent circuit optimization methods (Butz et al., 2026; Maene et al., 2025) are not directly comparable, as they operate on precompiled circuits. Methods including variable elimination and junction-tree inference scale exponentially in maximal

Table 1: Compilation and propagation statistics for representative BN2Os (top: fixed fan-in; bottom: QMR–DT–like). Circuit sizes are consistent with Theorem 2. Compilation time and propagation time $c(N)$ are mean $\pm$ 95% CI; propagation uses 100 runs, with peak RAM reported for both.

BN2O edges	Circuit edges	Compile time mean $\pm$ 95% CI (s)	Compile Peak RAM	Propagation $c(N)$ mean $\pm$ 95% CI (s)	Inference Peak RAM
200	2.70K	0.00555 $\pm$ 0.00006	12.79 MiB	0.00123 $\pm$ 0.00002	12.89 MiB
20K	270K	0.1554 $\pm$ 0.0009	29.41 MiB	0.1315 $\pm$ 0.0004	70.39 MiB
2M	27M	293.36 $\pm$ 0.39	1.64 GiB	15.058 $\pm$ 0.024	4.35 GiB
24M	324M	40,900 $\pm$ 2,300	20.16 GiB	191.9 $\pm$ 2.6	56.27 GiB
4.09K	45.2K	0.0221 $\pm$ 0.0002	14.63 MiB	0.0192 $\pm$ 0.0001	18.20 MiB
44K	483K	0.224 $\pm$ 0.002	33.80 MiB	0.214 $\pm$ 0.0003	78.41 MiB
444K	4.87M	4.319 $\pm$ 0.013	224.58 MiB	2.33 $\pm$ 0.003	783.18 MiB
13.5M	148M	1,650 $\pm$ 270	6.23 GiB	77.4 $\pm$ 0.12	22.48 GiB

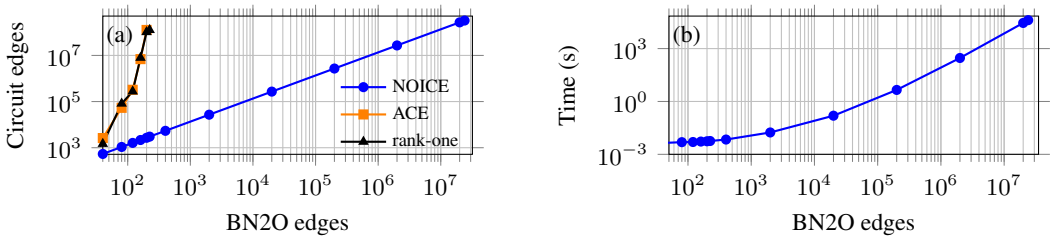


Figure 2: Compilation scaling for fixed fan-in-4 BN2Os of sizes 40 to 24M edges. (a) Compiled circuit edges versus BN2O edges on log–log scales; NOICE sizes are deterministic, while ACE and rank-one tensor decomposition points show the mean over 20 runs (95% CIs reported in Table 2 (Appendix B)). (b) NOICE mean compilation time versus BN2O edges (95% CIs are reported in Table 1 and Table 5 (Appendix B)).

clique size and are infeasible for QMR–DT (Jaakkola & Jordan, 1999; Ng & Jordan, 1999; Jordan et al., 1999; Li et al., 2011), where cliques exceed 150 nodes (Jaakkola & Jordan, 1999). We restrict evaluation to compilation-based methods that perform significant inference work offline and amortize it over many queries (Chavira & Darwiche, 2008; Darwiche, 2003; 2022), as comparisons with search-based approaches (Li et al., 2008) must be interpreted with care due to differing trade-offs and design goals (Chavira & Darwiche, 2008; Li et al., 2011). In all experiments, a fixed random seed of 1 was used, and confidence intervals (CIs) use the t-distribution. All source code, BN2O generators, raw results, and statistics are released to support full reproducibility.

Compilation size and time. For fixed fan-in-4 BN2Os, baseline compilation limits appear early: ACE compiles the 200-edge network but encounters a memory error at 224 edges, and the rank-one tensor construction fails at 228 edges, consistent with prior reports (Vomlel & Savický, 2008; Gogate & Domingos, 2010). Unlike NOICE, these approaches triangulate an undirected BN2O, inducing large cliques and memory exhaustion, reflecting the exponential dependence on Bayesian network treewidth (Darwiche, 2009). In contrast, NOICE compiles BN2Os up to 24M edges in the same environment, a five-order-of-magnitude improvement over baseline methods. For BN2Os where ACE and rank-one successfully compile, inference runtimes for $\omega(Q) \in \{0, 5, 10\}$ are reported in Table 3 (Appendix B). Table 1 (top) reports compilation and propagation statistics, showing circuit size scales linearly with BN2O size, consistent with Theorem 2, up to contributions from leak nodes. Circuit sizes are deterministic and reported exactly, while compilation times are measured over runs (mean $\pm$ 95% confidence intervals, with runs decreasing as network size increases). Propagation time $c(N)$ is the mean over 100 random circuit propagations with 95% CIs; peak RAM is reported separately for compilation and inference. Figure 2(a) shows circuit size versus BN2O size on log–log scales. NOICE (—●—) exhibits effectively linear growth: a log–log fit yields $\hat{\beta} = 1.0000$ (95% CI: [0.9994, 1.0007], $R^2 \approx 1$), consistent with Theorem 2. For clarity, ACE (—■—) and rank-one (—▲—) curves show means without error bars; corresponding 95% confidence intervals over 20 runs are reported in Table 2 (Appendix B). Compilation time is mildly superlinear (Figure 2(b)). Total times (compilation plus inference) are reported in Table 4 (Appendix B).

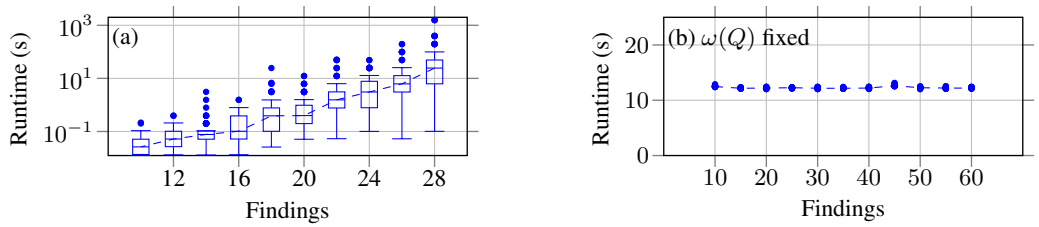


Figure 3: NOICE inference runtime versus findings in a representative 2K-edge BN2O. (a) Uncontrolled queries: boxplots (median, IQR, Tukey whiskers) summarize runtime over $n = 100$ randomly sampled queries per size; the y-axis is logarithmic. (b) Controlled queries with fixed overlap index $\omega(Q) = 10$: runtime versus $|Q|$ on a fixed y-axis scale. The same qualitative behavior is observed across all tested network sizes; Figure 8 (Appendix B) shows a 2M-edge BN2O.

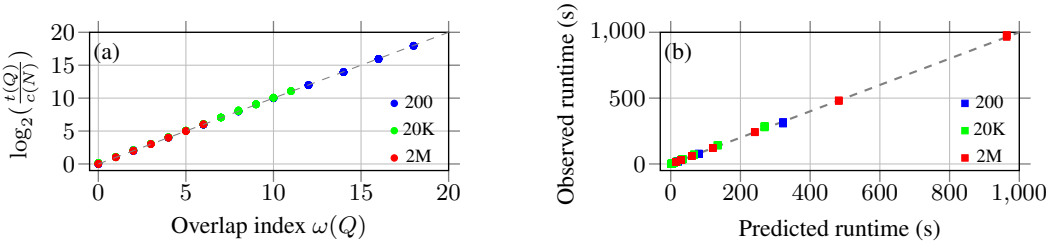


Figure 4: Empirical validation of the BN2O Overlap Law for representative fixed fan-in-4 BN2Os: (a) $\log_2(t(Q)/c(N))$ versus overlap index $\omega(Q)$; (b) predicted versus observed runtime.

Evidence size is not predictive. Figure 3(a) plots inference time versus findings in a representative 2K-edge BN2O. For each of ten query sizes, we generate $n = 100$ random queries and measure runtime. Boxplots summarize the distribution (median, IQR, and outliers), and the y-axis is logarithmic due to heavy-tailed runtimes. A dashed line connects median runtimes across query sizes. Across all sizes, runtimes exhibit variability in their interquartile ranges and heavy upper tails, indicating that queries with the same number of findings can differ widely in cost. Thus the number of findings alone does not predict inference runtime in NOICE circuits under random query generation.

Controlled overlap experiment. Figure 3(b) plots runtime versus evidence size for queries with overlap index $\omega(Q) = 10$, with a dashed line connecting median runtimes, while the number of findings varies from 15 to 60 (11 sizes) in the same 2K-edge BN2O. Additional observed leaves are selected to avoid introducing shared parents, preserving $\boldsymbol{S}$ and $\omega(Q)$. We generate $n = 100$ queries per evidence size. In contrast to the uncontrolled setting, runtime remains nearly constant as additional findings are observed. A linear fit yields a small slope, indicating negligible change in runtime relative to within-condition variation. Thus, once overlap is controlled, evidence size has at most a negligible practical effect on runtime. Complementary experiments fixing the number of findings while varying overlap are reported in Table 8 (Appendix B).

Structural overlap governs runtime. The controlled experiment above shows that inference time remains essentially constant when the overlap index $\omega(Q)$ is fixed, indicating that runtime depends on the combinatorial structure captured by $\omega(Q)$ rather than on evidence size. Eq. (4) predicts the runtime decomposition $t(Q) = c(N)\, 2^{\omega(Q)}$, which separates inference cost into a combinatorial factor determined by the query structure and a circuit-dependent propagation cost: $2^{\omega(Q)}$ counts configurations of the overlap set $\boldsymbol{S}$, and $c(N)$ is the time for a single propagation in a NOICE circuit of size N. Figure 4(a) plots $\log_2(t(Q)/c(N))$ versus $\omega(Q)$ for BN2Os with 200 (•), 20K (•), and 2M (•) edges, together with the reference line $\log_2(t(Q)/c(N)) = \omega(Q)$. Measurements lie closely along the line across all sizes. A linear fit $\log_2(t(Q)/c(N)) = \beta\,\omega(Q)$ yields $\beta \approx 1$ with 95% CIs tightly concentrated around 1 and $\mathrm{R}^2 > 0.9999$. Table 9 (Appendix B) reports runtime statistics.

Runtime prediction. We evaluate whether the BN2O Overlap Law $t(Q) = c(N)2^{\omega(Q)}$ predicts inference time. For each query we compute the predicted runtime $\hat{t}(Q) = c(N)2^{\omega(Q)}$ using the measured propagation cost $c(N)$. Figure 4(b) compares predicted (dashed) and observed runtimes

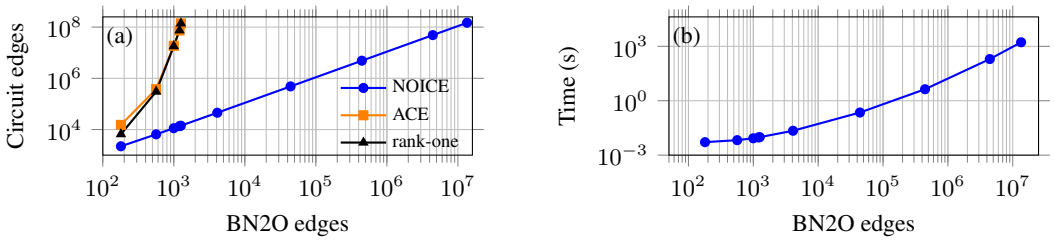


Figure 5: Compilation scaling for QMR–DT–like BN2Os of sizes 180 to 13.5M edges. (a) Compiled circuit edges versus BN2O edges. (b) NOICE compilation time versus BN2O edges.

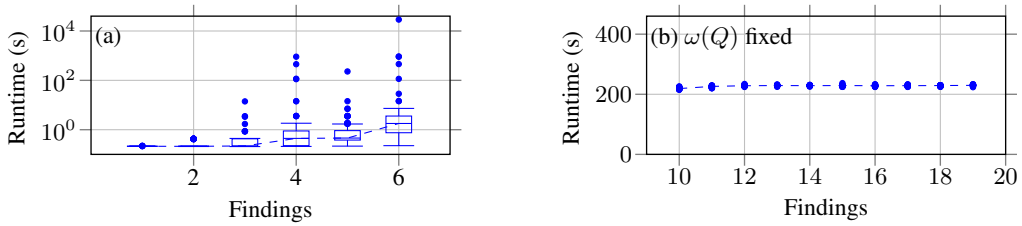


Figure 6: NOICE inference runtime versus findings in a representative 44K-edge QMR–DT–like BN2O: (a) boxplots; (b) fixed overlap $\omega(Q) = 10$. Dashed lines connect medians.

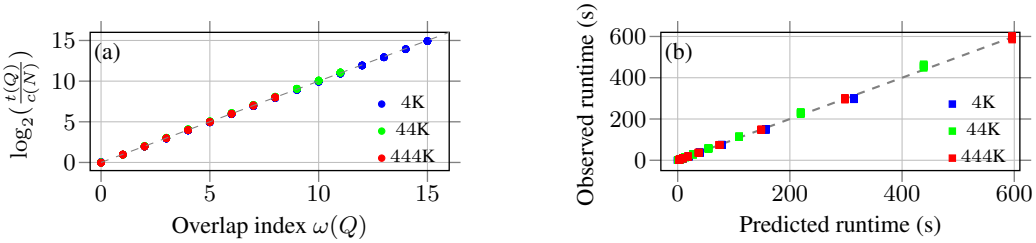


Figure 7: Empirical validation of the BN2O Overlap Law for QMR–DT–like BN2Os: (a) $\log_2(t(Q)/c(N))$ versus overlap index $\omega(Q)$ across sizes; (b) predicted versus observed runtime.

using the same queries, showing close agreement across tested networks and several orders of magnitude, with median relative prediction error 3.29% and maximum error 11.5%.

QMR–DT–like networks. To assess robustness beyond fixed fan-in, we evaluate BN2Os with QMR–DT–like structure (Shwe et al., 1991) under the same protocol. In our benchmark environment, ACE and the rank-one tensor construction fail at 1.33K edges, which is less than 3% of the QMR–DT scale ($\approx$ 45K edges). In contrast, NOICE compiles QMR–DT–like BN2Os with up to 13.5M edges, over four orders of magnitude larger under the same conditions. As a result, direct inference-time comparisons are limited to regimes far below practical scale. The same qualitative behavior is observed: circuit size scales linearly and compilation time mildly superlinearly (Figure 5), inference cost is not predicted by evidence size (Figure 6), and runtime follows the BN2O Overlap Law (Figure 7). Representative statistics are reported in Table 1 (bottom), confirming the same scaling behavior. Additional experimental and statistical details appear in Appendix C.

5 Conclusion

We showed that BN2Os, despite their multiply connected structure, admit an exact family-set factorization enabling scalable structural compilation. This factorization yields NOICE circuits whose size grows linearly with BN2O size and induces the BN2O Overlap Law, $t(Q) = c(N)2^{\omega(Q)}$. Thus, inference cost is governed by the overlap index $\omega(Q)$ rather than network size. Experiments on BN2Os with up to 24M edges validate this law and show compilation scaling more than five orders of magnitude beyond existing general-purpose approaches. Extending NOICE to multi-layer noisy-OR networks is a promising direction for future work (Chavira et al., 2005).

AI USE STATEMENT

In this work, we did not use generative AI tools for any of the tasks requiring disclosure; those tasks were either performed entirely by the authors or were not applicable to this work. We used generative AI tools only to edit parts of the manuscript to improve readability and language. All AI-assisted text was carefully reviewed, revised, and verified by the authors to ensure that it accurately reflects the authors' intended meaning and the scientific content of the work. Generative AI tools were not used to develop the methodology, design or conduct the experiments, formulate mathematical claims, produce proofs, interpret results, or generate scientific claims. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The paper states the assumptions for its theoretical results in the main text, proves key results (including Theorem 2 on linear compilation size) in the main text, and provides complete formal proofs in the appendix, with numbered statements and cross-references throughout (Sections 2 and 3; Appendix A). The paper specifies the BN2O families studied, network sizes, query generation procedures, hardware environment, timeout policy, and reported metrics needed to reproduce the main experimental claims (Section 4; Appendices B, C, and D). Additional experimental details and statistical summaries are provided in the appendices for both fixed fan-in and QMR–DT–like networks (Appendices B and C). Appendix D contains experimental setup for full reproducibility. We provide access to all anonymized code, data generation procedures, and experimental results in the supplemental material, enabling full reproducibility and verify the main findings (Section 4; Appendices B, C, and D).

REFERENCES

D. Agrawal, Y. Pote, and K.S. Meel. Partition function estimation: A quantitative study. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 4276–4285, 2021. doi: 10.24963/ijcai.2021/587.

I. A. Auzina and J. M. Tomczak. Approximate Bayesian computation for discrete spaces. *Entropy*, 23(3):312, 2021. doi: 10.3390/e23030312.

C. Butz, A. Santoscoy-Rivero, and C. Lewis. Paths not taken: Structure-based pruning in PSDD learning and inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 19799–19807, 2026.

M. Chavira and A. Darwiche. On probabilistic inference by weighted model counting. *Artificial Intelligence*, 172(6):772–799, 2008. doi: 10.1016/j.artint.2007.11.002.

M. Chavira, D. Allen, and A. Darwiche. Exploiting evidence in probabilistic inference. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pp. 112–127, 2005.

Y. Choi, A. Vergari, and G. Van den Broeck. Probabilistic circuits: A unifying framework for tractable probabilistic models, 2020. URL `http://starai.cs.ucla.edu/papers/ProbCirc20.pdf`.

A. Darwiche. A differential approach to inference in Bayesian networks. *Journal of the ACM*, 50 (3):280–305, 2003. doi: 10.1145/765568.765570.

A. Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, 2009.

A. Darwiche. Tractable boolean and arithmetic circuits, 2022. URL `https://arxiv.org/abs/2202.02942`.

A. Dedieu, G. Zhou, D. George, and M. Lázaro-Gredilla. Learning noisy-OR Bayesian networks with max-product belief propagation. In *Proceedings of the International Conference on Machine Learning*, pp. 7426–7448. PMLR, 2023.

V. Gogate and P. Domingos. Formula-based probabilistic inference. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pp. 210–219, 2010.

D. Heckerman. A tractable inference algorithm for diagnosing multiple diseases. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pp. 163–171, 1990.

T. S. Jaakkola and M. I. Jordan. Variational probabilistic inference and the QMR-DT network. *Journal of Artificial Intelligence Research*, 10:291–322, 1999.

M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999. doi: 10.1023/A:1007665907178.

W. Li, P. Poupart, and P. van Beek. Exploiting causal independence using weighted model counting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 337–343, 2008.

W. Li, P. Poupart, and P. van Beek. Exploiting structure in weighted model counting approaches to probabilistic inference. *Journal of Artificial Intelligence Research*, 40(1):729–765, 2011.

J. Maene, V. Derkinderen, and P. Zuidberg Dos Martires. KLay: Accelerating arithmetic circuits for neurosymbolic AI. In *Proceedings of the International Conference on Learning Representations*, 2025.

A. Y. Ng and M. I. Jordan. Approximate inference algorithms for two-layer Bayesian networks. *Advances in Neural Information Processing Systems*, 12:533–539, 1999.

K. G. Olesen, U. B. Kjærulff, F. Jensen, F. V. Jensen, B. Falck, S. Andreassen, and S. K. Andersen. A MUNIN network for the median nerve: A case study on loops. *Applied Artificial Intelligence*, 3(2–3):385–403, 1989.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

J. G. Richens, C. M. Lee, and S. Johri. Improving the accuracy of medical diagnosis with causal machine learning. *Nature Communications*, 11(1):3923, 2020.

M. Shwe, B. Middleton, D. Heckerman, M. Henrion, E. Horvitz, H. Lehmann, and G. Cooper. Probabilistic diagnosis using a reformulation of the INTERNIST-1/QMR knowledge base. *Methods of Information in Medicine*, 30(4):241–255, 1991.

UCLA Automated Reasoning Group. ACE: Arithmetic circuit evaluator. Software and documentation, 2015. Available at `http://reasoning.cs.ucla.edu/ace`. Accessed: 2026-04-10.

A. Vergari, Y. Choi, A. Liu, S. Teso, and G. Van den Broeck. A compositional atlas of tractable circuit operations for probabilistic inference. In *Advances in Neural Information Processing Systems*, pp. 13189–13201, 2021.

J. Vomlel and P. Savický. Arithmetic circuits of the noisy-or models. In *Proceedings of the European Workshop on Probabilistic Graphical Models*, pp. 297–304, 2008.

B. Wang and M. Kwiatkowska. Symbolic causal inference via operations on probabilistic circuits. In *Advances in Neural Information Processing Systems*, 2022.

B. Wang and M. Kwiatkowska. Compositional probabilistic and causal inference using tractable circuit models. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 9488–9498. PMLR, 2023.

Y. Yang, G. Gala, and R. Peharz. Bayesian structure scores for probabilistic circuits. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 563–575. PMLR, 2023.

L. Yin and H. Zhao. On the expressive power of tree-structured probabilistic circuits. In *Advances in Neural Information Processing Systems*, pp. 73830–73857, 2024.

H. Zhang, B. Wang, M. Arenas, and G. Van den Broeck. Restructuring tractable probabilistic circuits. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 2566–2574. PMLR, 2025.

A PROOFS

Proof of Theorem 1

Proof. In a BN2O, the joint distribution factorizes as

$$P(\boldsymbol{X}, \boldsymbol{Y}) = \prod_{Z \in \boldsymbol{X} \cup \boldsymbol{Y}} \frac{P(F_Z)}{P(\pi_Z)}. \tag{5}$$

Since every parent set π_Z consists only of roots, and the roots are mutually independent, Eq. (5) becomes

$$\begin{aligned} P(\boldsymbol{X}, \boldsymbol{Y}) &= \prod_{Z \in \boldsymbol{X} \cup \boldsymbol{Y}} \frac{P(F_Z)}{\prod_{X_i \in \pi_Z} P(X_i)} \\ &= \left(\prod_{X_i \in \boldsymbol{X}} P(F_{X_i}) \right) \left(\prod_{Y_j \in \boldsymbol{Y}} \frac{P(F_j)}{\prod_{X_i \in \pi_j} P(X_i)} \right) \\ &= \frac{\left(\prod_{X_i \in \boldsymbol{X}} P(X_i) \right) \left(\prod_{Y_j \in \boldsymbol{Y}} P(F_j) \right)}{\prod_{Y_j \in \boldsymbol{Y}} \prod_{X_i \in \pi_j} P(X_i)}. \end{aligned} \tag{6}$$

For each root X_i, the factor $P(X_i)$ appears in the denominator once for every leaf Y_j such that $X_i \in \pi_j$, i.e., exactly d_i times. Hence

$$\prod_{Y_j \in \boldsymbol{Y}} \prod_{X_i \in \pi_j} P(X_i) = \prod_{X_i \in \boldsymbol{X}} P(X_i)^{d_i}. \tag{7}$$

Substituting Eq. (7) into Eq. (6) and cancelling gives Eq. (2). □

We prove Theorem 3 by first establishing the correctness of the root and family-set subcircuits.

Lemma 1 (Root Subcircuit Correctness)**.** *Let $X_i \in \boldsymbol{X}$ be a BN2O root with marginal $P(X_i)$. The NOICE root subcircuit $\mathcal{C}(X_i)$ in Figure 1(a) evaluates to $P(x_i)$ when the indicator for x_i is set, to $P(\bar{x}_i)$ when the indicator for $\bar{x}_i$ is set, and to 1 when both indicators are set.*

Proof. During NOICE compilation, each root X_i is treated independently together with its marginal $P(X_i)$. The single-node network consisting solely of X_i with marginal $P(X_i)$ is a valid Bayesian network. Applying the variable-elimination-based arithmetic circuit compilation algorithm of Darwiche Darwiche (2003) produces the canonical arithmetic circuit for $P(X_i)$ under indicator semantics, exactly the root subcircuit template $\mathcal{C}(X_i)$ in Figure 1(a). The claims follow. □

Lemma 2 (Family-Set Subcircuit Correctness)**.** *Let $Y_j \in \boldsymbol{Y}$ be a BN2O leaf with parent set $\pi_j \subseteq \boldsymbol{X}$, and let $F_j = \{Y_j\} \cup \pi_j$ denote its family. Let $\mathcal{C}(F_j)$ be the NOICE family-set subcircuit in Figure 1(b). Consider any indicator setting for the variables in F_j such that, for each variable, at least one of its indicators is set to one. If every variable in F_j has both indicators set to one, then $\mathcal{C}(F_j)$ evaluates to 1. If every variable in F_j has exactly one indicator set to one, then $\mathcal{C}(F_j)$ evaluates to $P(f_j)$ for the induced complete instantiation f_j of F_j. Otherwise, let $\boldsymbol{Z} \subset F_j$ denote the variables with exactly one indicator set to one, and let $\boldsymbol{z}$ be the induced assignment. Then $\mathcal{C}(F_j)$ evaluates to $P(\boldsymbol{Z} = \boldsymbol{z})$.*

Proof. Fix a leaf Y_j with parent set $\pi_j = \{X_1, \ldots, X_k\}$ and family $F_j = \{Y_j\} \cup \pi_j$. The family-set marginal is

$$P(F_j) = P(Y_j, \pi_j) = P(Y_j \mid \pi_j) \prod_{X_i \in \pi_j} P(X_i).$$

Vomlel and Savický Vomlel & Savický (2008) show that a noisy-OR gate with Boolean parents admits a linear-size arithmetic circuit under indicator semantics. In our notation, the corresponding multilinear polynomial is

$$\lambda_{y_j} \prod_{X_i \in \pi_j} \left(\lambda_{\bar{x}_i}\theta_{\bar{x}_i} + \lambda_{x_i}\theta_{x_i}\right) + (\lambda_{\bar{y}_j} - \lambda_{y_j}) \prod_{X_i \in \pi_j} \left(\lambda_{\bar{x}_i}\theta_{\bar{x}_i} + \lambda_{x_i}\theta_{x_i} q_i^j\right),$$

exactly the family-set subcircuit $\mathcal{C}(F_j)$ in Figure 1(b).

Under standard indicator semantics, an arithmetic circuit evaluates to the probability of the event specified by the indicators. We verify the three cases.

If every variable in F_j has both indicators set to one, then no variable in the family is fixed and the circuit sums over all configurations of F_j. Hence

$$\mathcal{C}(F_j) = \sum_{f_j} P(f_j) = 1.$$

If every variable in F_j has exactly one indicator set to one, then the indicators specify a complete instantiation f_j of the family, and the circuit evaluates directly to

$$\mathcal{C}(F_j) = P(f_j).$$

Otherwise, let $\boldsymbol{Z} \subset F_j$ denote the variables with exactly one indicator set to one and let $\boldsymbol{z}$ be the induced assignment. Every variable in $F_j \setminus \boldsymbol{Z}$ has both indicators set to one and is therefore marginalized out by the circuit. Consequently,

$$\mathcal{C}(F_j) = \sum_{f_j \sim \boldsymbol{z}} P(f_j) = P(\boldsymbol{Z} = \boldsymbol{z}),$$

where $f_j \sim \boldsymbol{z}$ denotes assignments f_j consistent with $\boldsymbol{z}$ on $\boldsymbol{Z}$. This establishes the three claims. □

Proof of Theorem 3

Proof. By Lemma 1, each root subcircuit $\mathcal{C}(X_i)$ evaluates correctly under indicator semantics. By Lemma 2, each family-set subcircuit $\mathcal{C}(F_j)$ evaluates to the probability of the event determined by the indicator setting.

Algorithm 1 constructs $\mathcal{C}$ as

$$\mathcal{C} = \frac{\prod_{Y_j \in \boldsymbol{Y}} \mathcal{C}(F_j)}{\prod_{X_i \in \boldsymbol{X}} \mathcal{C}(X_i)^{d_i - 1}},$$

which matches the factorization of $P(\boldsymbol{X}, \boldsymbol{Y})$ given in Theorem 1. For any complete instantiation $(\boldsymbol{x}, \boldsymbol{y})$, exactly one indicator per variable is set to one, so each $\mathcal{C}(F_j)$ and $\mathcal{C}(X_i)$ evaluates to the corresponding probability term. Therefore, $\mathcal{C}$ evaluates to $P(\boldsymbol{x}, \boldsymbol{y})$.

Algorithm 2 computes $P(\boldsymbol{Z} = \boldsymbol{z})$ by explicitly summing over assignments to the non-queried shared roots $\boldsymbol{S}$ in line 9, while all other variables are handled by indicator evaluation. Indicator settings are global across the circuit, so each variable has a single pair of indicators whose values are consistent across all subcircuits during propagation. By Lemma 2, each family-set subcircuit evaluates to the marginal over the variables whose indicators are fixed, and variables with both indicators set to one are thereby marginalized.

Any non-queried root $X_i \notin \boldsymbol{S}$ has at most one observed child. For such a non-queried root X_i, both indicators remain set to one in every propagation. By Lemma 2, each occurrence of X_i in a family-set subcircuit is therefore marginalized locally, conditional on any other variables in that family whose indicators are fixed. Thus, X_i appears in at most one family-set subcircuit corresponding to an observed leaf, and is therefore eliminated locally within that subcircuit without introducing dependencies across multiple factors involving observed variables, while all other occurrences are marginalized within their respective family-set subcircuits. Therefore, the only variables whose occurrences introduce dependencies across multiple factors involving observed variables are the non-queried shared roots $\boldsymbol{S}$, and enumerating over $\boldsymbol{S}$ yields the marginal $P(\boldsymbol{Z} = \boldsymbol{z})$. □

Table 2: Compilation-size statistics for non-deterministic baseline methods over 20 runs per network size. We report mean compiled circuit size with 95% CIs.

BN2O edges	Circuit edges (mean ± 95% CI) ACE	rank-one
40	2.63K ±0.01K	1.47K ±0.01K
80	53.5K ±0.03K	81.6K ±0.6K
120	324K ±0.7K	276K ±2.7K
160	6.91M ±0.03M	7.91M ±0.5M
200	127M ±0.01M	102M ±0.7M
224	Compilation failure	126M ±0.01M
228	Compilation failure	Compilation failure

Proof of Theorem 4

Proof. Algorithm 2 first computes the overlap set $\boldsymbol{S}$ on line 4. If $\boldsymbol{S} = \emptyset$, the algorithm performs one propagation and returns its value, and since $|\boldsymbol{S}| = 0$ this equals $2^{|\boldsymbol{S}|}$ propagations.

Otherwise, $\boldsymbol{S}$ is a set of non-queried shared root variables, so $|\mathrm{dom}(\boldsymbol{S})| = 2^{|\boldsymbol{S}|}$. The **for** loop enumerates each assignment $\boldsymbol{s} \in \mathrm{dom}(\boldsymbol{S})$, performing exactly one circuit propagation per assignment. Therefore Algorithm 2 performs exactly $2^{|\boldsymbol{S}|}$ circuit propagations.

Finally, since $\boldsymbol{S} \subseteq \boldsymbol{X}$, we have $|\boldsymbol{S}| \leq r$, implying worst-case complexity $O(2^r)$ circuit propagations. □

B Fixed fan-in BN2Os: detailed experimental and statistical results

This appendix provides detailed experimental and statistical results supporting the fixed fan-in BN2O results in Section 4.

Compilation size and time (baseline methods). Table 2 reports compilation-size statistics for ACE and rank-one. Variability arises from non-deterministic compilation across runs. 95% CIs are reported over 20 runs per configuration.

Inference runtime (baseline methods). Table 3 reports inference runtimes for ACE, rank-one, and NOICE on BN2Os where ACE and rank-one successfully compile, for $\omega(Q) \in \{0, 5, 10\}$. Entries marked — indicate inference failure due to memory limits.

Total compilation and inference runtime (baseline methods). Table 4 reports total compilation and inference runtimes for ACE, rank-one, and NOICE for fixed fan-in-4 BN2Os with $\omega(Q) \in \{0, 5, 10\}$. For BN2Os with 200 or more edges, ACE and rank-one encounter inference or compilation failures.

Compilation size and time. For fixed fan-in-4 BN2Os, baseline compilation limits appear early: ACE compiles the 200-edge network but encounters a memory error at 224 edges, and the rank-one tensor construction fails at 228 edges, consistent with prior reports Vomlel & Savický (2008); Gogate & Domingos (2010). Unlike NOICE, these approaches triangulate an undirected representation of the BN2O, inducing large cliques and causing memory exhaustion, reflecting the exponential dependence on Bayesian network treewidth Darwiche (2009). In contrast, NOICE compiles BN2Os up to 24M edges in the same environment, a five-order-of-magnitude improvement over baseline methods. Table 5 reports compilation and propagation statistics for BN2Os, showing that circuit size scales linearly with BN2O size consistent with Theorem 2, up to contributions from leak nodes. Circuit sizes are deterministic and reported exactly, while compilation times are measured over runs (mean ±95% CIs with runs decreasing with network size). Propagation time $c(N)$ is computed as the mean over 100 random circuit propagations with 95% CIs, and peak RAM denotes maximum memory usage.

Table 3: Inference runtimes (s) for fixed fan-in BN2Os, reported as mean ± 95% CI over 20 runs per network size, for $\omega(Q) \in \{0, 5, 10\}$.

BN2O edges	Overlap Index $\omega(Q)$	Inference time mean ± 95% CI (s) ACE	rank-one	NOICE
40	0	0.00195 ± 0.00002	0.00146 ± 0.00001	0.00027 ± 0.00001
	5	0.00197 ± 0.00007	0.00173 ± 0.00009	0.00789 ± 0.00004
	10	0.00199 ± 0.00006	0.00155 ± 0.00005	0.2427 ± 0.0007
80	0	0.00926 ± 0.00021	0.00971 ± 0.00047	0.00052 ± 0.00005
	5	0.00953 ± 0.00023	0.00978 ± 0.00017	0.03284 ± 0.00017
	10	0.0094 ± 0.0004	0.00944 ± 0.00011	0.75 ± 0.11
120	0	0.01385 ± 0.00021	0.01314 ± 0.00049	0.00077 ± 0.00006
	5	0.01387 ± 0.00019	0.01282 ± 0.00043	0.02314 ± 0.00009
	10	0.01409 ± 0.00025	0.01304 ± 0.00039	0.718 ± 0.003
160	0	0.072 ± 0.004	0.068 ± 0.002	0.00102 ± 0.00006
	5	0.077 ± 0.004	0.072 ± 0.003	0.03072 ± 0.00026
	10	0.082 ± 0.005	0.074 ± 0.004	0.965 ± 0.005
200	0	Inference failure	Inference failure	0.00127 ± 0.00006
	5	Inference failure	Inference failure	0.03932 ± 0.00021
	10	Inference failure	Inference failure	1.211 ± 0.008

Table 4: Fixed fan-in total compilation and inference times (s), mean ± 95% CI (20 runs), $\omega(Q) \in \{0, 5, 10\}$; totals equal compilation plus the inference times in Table 3.

BN2O edges	$\omega(Q)$	ACE	rank-one	NOICE
40	0	0.05055 ± 0.0009	0.03886 ± 0.0002	0.00471 ± 0.00005
	5	0.05057 ± 0.0009	0.03913 ± 0.0002	0.0126 ± 0.00006
	10	0.05059 ± 0.0009	0.03895 ± 0.0002	0.2553 ± 0.0007
80	0	0.08176 ± 0.0009	0.07471 ± 0.0006	0.00545 ± 0.0001
	5	0.08203 ± 0.0009	0.07478 ± 0.0004	0.03777 ± 0.0002
	10	0.0819 ± 0.0009	0.07444 ± 0.0004	0.755 ± 0.110
120	0	0.13185 ± 0.001	0.10814 ± 0.001	0.00574 ± 0.00008
	5	0.13187 ± 0.001	0.10782 ± 0.001	0.02811 ± 0.0001
	10	0.13209 ± 0.001	0.10804 ± 0.001	0.72297 ± 0.003
160	0	0.850 ± 0.009	0.892 ± 0.042	0.00633 ± 0.00008
	5	0.855 ± 0.009	0.896 ± 0.042	0.03603 ± 0.0002
	10	0.860 ± 0.010	0.898 ± 0.042	0.97031 ± 0.005
200	0	Inference failure	Inference failure	0.00682 ± 0.00008
	5	Inference failure	Inference failure	0.04487 ± 0.0002
	10	Inference failure	Inference failure	1.21655 ± 0.008
224	0	Compilation failure	Inference failure	0.00716 ± 0.0002
	5	Compilation failure	Inference failure	0.0481 ± 0.0001
	10	Compilation failure	Inference failure	1.36654 ± 0.0038

Evidence size is not predictive. Figure 8 reproduces Figure 3 on a 2M-edge BN2O, including dashed lines connecting median runtimes. Table 6 reports summary statistics for uncontrolled overlap experiments.

Controlled overlap experiment. Table 7 reports summary statistics. Across the full range of $|Q|$, relative IQR remains below 1.3%, indicating negligible dependence on evidence size under fixed overlap.

Controlled findings experiment. Table 8 reports the complementary controlled experiment to Figure 3(b), fixing the number of observed findings while varying the overlap index on a representative fixed fan-in-4 BN2O with 2K edges ($|Q| = 20$). Across these controlled queries, inference time follows the exponential growth predicted by the BN2O Overlap Law, demonstrating that, for a fixed number of observed findings, inference runtime increases exponentially with the overlap index. To-

Table 5: NOICE compilation and propagation statistics for fixed fan-in BN2Os. Circuit sizes are consistent with Theorem 2. Compilation time and propagation time $c(N)$ are mean $\pm$ 95% CI; propagation uses 100 runs, and peak RAM usage is reported for both.

BN2O edges	Circuit edges	Compile time mean $\pm$95% CI (s)	Compile Peak RAM	Propagation $c(N)$ mean $\pm$95% CI (s)	Inference Peak RAM
40	538	0.00444 ± 0.00005	12.25 MiB	0.000260 ± 0.000002	12.45 MiB
80	1.08K	0.00493 ± 0.00014	12.66 MiB	0.00051 ± 0.00001	12.63 MiB
120	1.62K	0.00497 ± 0.00006	12.72 MiB	0.000727 ± 0.000003	12.70 MiB
160	2.16K	0.00531 ± 0.00009	12.63 MiB	0.000983 ± 0.000005	12.73 MiB
200	2.70K	0.00555 ± 0.00006	12.79 MiB	0.00123 ± 0.00002	12.89 MiB
224	3.02K	0.00572 ± 0.00006	12.76 MiB	0.001364 ± 0.000006	13.02 MiB
400	5.40K	0.00688 ± 0.00007	13.05 MiB	0.00248 ± 0.00001	13.61 MiB
2K	27K	0.01741 ± 0.00018	14.55 MiB	0.01244 ± 0.00003	16.82 MiB
20K	270K	0.1554 ± 0.0009	29.41 MiB	0.1315 ± 0.0004	70.39 MiB
200K	2.70M	4.513 ± 0.012	184.41 MiB	1.4451 ± 0.0017	528.86 MiB
2M	27M	293.36 ± 0.39	1.64 GiB	15.058 ± 0.024	4.35 GiB
20M	270M	$28,500 \pm 1,400$	16.45 GiB	156.27 ± 0.21	53.53 GiB
24M	324M	$40,900 \pm 2,300$	20.16 GiB	191.9 ± 2.6	56.27 GiB

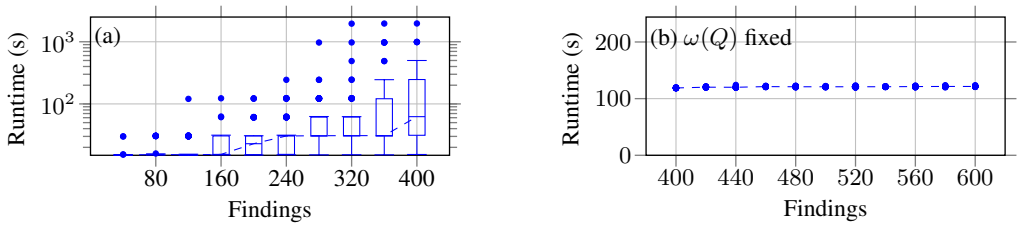


Figure 8: NOICE inference runtime versus findings in a representative 2M-edge fixed fan-in BN2O: (a) boxplots of runtime; (b) queries with fixed overlap index $\omega(Q) = 3$. This experiment replicates Figure 3 at larger scale.

gether, these experiments show that the number of observed findings and the overlap index capture distinct structural properties: fixing either quantity does not determine the other, while the BN2O Overlap Law accurately predicts the resulting inference time.

Structural overlap governs runtime. Table 9 reports fit statistics for representative networks spanning several orders of magnitude. We evaluate the scaling law $t(Q) = c(N)\, 2^{\omega(Q)}$ by fitting the linear model

$$\log_2\left(\frac{t(Q)}{c(N)}\right) = \beta\, \omega(Q)$$

using ordinary least squares without intercept. Across networks, slope estimates are close to 1, with 95% CIs tightly concentrated around 1 and coefficients of determination (R^2) exceeding 0.9999. Variability reflects random sampling of queries at fixed overlap levels. All other network sizes exhibit the same behavior, with slope estimates within a small fraction of 1 and similarly tight CIs.

Runtime prediction accuracy. Table 10 reports prediction errors across representative fixed fan-in BN2Os.

C QMR–DT–LIKE BN2OS: EXPERIMENTAL AND STATISTICAL RESULTS

This appendix presents experimental and statistical results for QMR–DT–like BN2Os supporting the results in Section 4.

C.1 EXPERIMENTAL RESULTS

This subsection reports detailed experimental results for the QMR–DT–like BN2Os discussed in Section 4. Network generation and experimental setup follow Section 4. These networks are generated using a truncated power-law leaf fan-in distribution with preferential parent attachment, pro-

Table 6: Uncontrolled-overlap queries for a representative 2M-edge fixed fan-in BN2O. For each query size $|Q|$, we sample 100 random queries without constraining $\omega(Q)$. Reported statistics summarize highly skewed runtime distributions. The relative tail is defined as $100 \times (90\text{th pct.} - \text{median})/\text{median}$, where the median denotes the typical runtime.

Findings $\lvert Q\rvert$	Median (s)	IQR (s)	90th pct. (s)	Tail relative to median (%)
80	15.32	0.21	17.37	13.40
160	15.41	15.22	30.86	100.20
240	30.48	15.58	61.52	101.80
320	30.86	30.98	122.42	296.70
400	62.36	215.60	494.85	693.60

Table 7: Controlled-overlap queries for a representative 2M-edge fixed fan-in BN2O with $\omega(Q) = 3$. Reported statistics summarize runtime variability as a function of evidence size $|Q|$, showing near-constant runtime under fixed overlap. Relative IQR is defined as $100 \times \text{IQR}/\text{median}$.

Findings $\lvert Q\rvert$	Median (s)	IQR (s)	Relative IQR (%)
400	118.83	0.59	0.50
440	120.04	0.61	0.51
480	120.95	0.96	0.79
520	120.95	1.52	1.25
560	121.30	1.24	1.03
600	121.48	0.68	0.56

ducing hub parents and overlap patterns similar to diagnostic networks. Each leaf includes a leak parameter, which is not counted in BN2O edge totals but is included in compiled circuit size counts. The benchmark suite ranges from 180 up to 13.5M BN2O edges.

Compilation size and time. Table 11 reports compilation and propagation statistics for QMR–DT–like BN2Os. Circuit sizes exactly match the closed-form expressions of Theorem 2, up to contributions from leak nodes. In our benchmark environment, both the ACE arithmetic circuit compiler and the rank-one tensor construction compile the QMR–DT–like network 20–148–1.26K but encounter memory errors at the slightly larger 21–154–1.33K (Table 13). In contrast, NOICE compiles QMR–DT–like BN2Os up to 13.5M edges under the same conditions (network size 172K–1.2M–13.5M, roots–leaves–edges), representing four orders of magnitude larger networks than the baselines. Figure 5(a) plots compiled circuit size versus BN2O size for NOICE (—●—), ACE (—■—), and the rank-one tensor construction (—▲—) on log–log scales. The template-based construction yields NOICE circuits whose edge count grows linearly with BN2O edges, as characterized by Theorem 2; compilation time likewise scales superlinearly with network size (Figure 5(b)).

Evidence size is not predictive. Figure 6(a) shows the same experiment on a 44K-edge QMR–DT–like BN2O, where runtime again varies widely for queries with similar evidence size, confirming that inference cost depends on structural overlap rather than the number of observations.

Controlled overlap experiment. Figure 6(b) shows that when the overlap index $\omega(Q)$ is held fixed while the number of observed findings varies, runtime remains essentially constant, confirming that inference cost is governed by the overlap index.

Controlled findings experiment. Table 12 reports the complementary controlled experiment to Figure 6(b), fixing the number of observed findings while varying the overlap index on a representative QMR–DT–like BN2O with approximately 44K edges ($|Q| = 40$). Across these controlled queries, inference time follows the exponential growth predicted by the BN2O Overlap Law, demonstrating that, for a fixed number of observed findings, inference runtime increases exponentially with the overlap index. Together, these experiments show that the number of observed findings and the overlap index capture distinct structural properties: fixing either quantity does not determine the other, while the BN2O Overlap Law accurately predicts the resulting inference time.

Structural overlap governs runtime. Figure 7(a) shows that runtimes follow the exponential law $t(Q) = c(N)2^{\omega(Q)}$ across QMR–DT–like networks with 4K, 44K, and 444K edges.

Table 8: Controlled findings experiment on a representative fixed fan-in-4 BN2O (2K edges, $|Q| = 20$). Runtime is reported as mean $\pm$ 95% confidence interval over randomly sampled queries for each overlap index.

$\omega(Q)$	Inference time mean $\pm$95% CI (s)
0	0.0246 ± 0.0003
1	0.0491 ± 0.0001
2	0.0979 ± 0.0004
3	0.195 ± 0.001
4	0.284 ± 0.043
5	0.387 ± 0.001
6	0.778 ± 0.003
7	1.57 ± 0.01
8	3.12 ± 0.01
9	6.22 ± 0.01
10	12.5 ± 0.1
11	24.9 ± 0.1
12	49.7 ± 0.1
13	99.3 ± 0.2
14	198.5 ± 0.3

Table 9: Linear regression results for $\log_2(t(Q)/c(N))$ versus $\omega(Q)$ on representative fixed fan-in BN2Os. Slope estimates $\hat{\beta}$ are reported with 95% confidence intervals. Fits exhibit near-perfect linearity ($R^2 \approx 1$).

BN2O edges	n	$\omega(Q)$ range	$\hat{\beta}$ (95% CI)	R^2
200	200	[0, 18]	0.99715 ± 0.00038	0.99999
20K	180	[0, 11]	1.0086 ± 0.0011	0.99994
2M	35	[0, 6]	1.0015 ± 0.0013	0.99999
All networks	415	[0, 18]	1.0000 ± 0.00065	0.99996

Runtime prediction. Figure 7(b) shows close agreement between predicted and observed runtimes using $\hat{t}(Q) = c(N)2^{\omega(Q)}$, confirming the accuracy of the runtime model for QMR–DT–like networks.

C.2 Detailed statistical results

This subsection reports detailed statistical results for QMR–DT–like BN2Os corresponding to Appendix B.

Compilation size and time (baseline methods). Table 13 reports compilation-size statistics for ACE and rank-one.

Total compilation and inference runtime (baseline methods). Table 14 reports total compilation and inference runtime for ACE, rank-one and NOICE for QMR–DT–like BN2Os with $\omega(Q) \in \{0, 4, 8\}$. The largest QMR–DT–like BN2O for which baselines ACE and the rank-one tensor construction complete total time has 998 edges. By comparison, the compilation and inference results reported show that NOICE completes total time on QMR–DT–like BN2Os with up to 13.5M edges, over four orders of magnitude larger under the same conditions.

Evidence size is not predictive. Table 15 reports summary statistics for uncontrolled overlap experiments on a 44K-edge network.

Controlled overlap experiments. Table 16 reports summary statistics for controlled overlap experiments on a 44K-edge network.

Structural overlap governs runtime. Table 17 reports fit statistics for QMR–DT–like networks.

Runtime prediction accuracy. Table 18 reports prediction errors for QMR–DT–like networks.

Table 10: Prediction accuracy of $\hat{t}(Q) = c(N)2^{\omega(Q)}$ on representative fixed fan-in BN2Os. Errors are reported as relative error $100 \times |t(Q) - \hat{t}(Q)|/t(Q)$.

BN2O edges	Runs	Median relative error (%)	IQR (%)	Max relative error (%)
200	200	2.22	2.31	10.9
20K	180	4.45	1.18	11.5
2M	35	0.44	0.90	2.97
All networks	415	3.29	3.03	11.5

Table 11: NOICE compilation and propagation statistics for QMR–DT–like BN2Os. Circuit sizes are consistent with Theorem 2. Compilation time and propagation time $c(N)$ are mean $\pm$ 95% CI; propagation uses 100 runs, and peak RAM usage is reported for both.

BN2O edges	Circuit edges	Compile time mean $\pm$ 95% CI (s)	Compile Peak RAM	Propagation $c(N)$ mean $\pm$ 95% CI (s)	Inference Peak RAM
180	2.22K	0.00522 $\pm$ 0.00018	12.50 MiB	0.000970 $\pm$ 0.000011	12.77 MiB
564	6.49K	0.00669 $\pm$ 0.00008	12.93 MiB	0.002710 $\pm$ 0.000007	13.29 MiB
998	11.3K	0.008535 $\pm$ 0.000041	13.05 MiB	0.004753 $\pm$ 0.000010	13.95 MiB
1.20K	13.5K	0.00951 $\pm$ 0.00011	13.25 MiB	0.005654 $\pm$ 0.000013	14.07 MiB
1.26K	14.1K	0.00981 $\pm$ 0.00010	13.27 MiB	0.00590 $\pm$ 0.00002	14.25 MiB
1.33K	15K	0.0217 $\pm$ 0.0002	13.32 MiB	0.01278 $\pm$ 0.00003	14.31 MiB
4.09K	45.2K	0.02211 $\pm$ 0.00017	14.63 MiB	0.01917 $\pm$ 0.00005	18.20 MiB
44K	483K	0.2241 $\pm$ 0.0018	33.80 MiB	0.2141 $\pm$ 0.0003	78.41 MiB
444K	4.87M	4.319 $\pm$ 0.013	224.58 MiB	2.3291 $\pm$ 0.0026	783.18 MiB
4.43M	48.6M	199.12 $\pm$ 0.52	2.06 GiB	24.35 $\pm$ 0.03	6.34 GiB
13.5M	148M	1,650 $\pm$ 270	6.22 GiB	77.35 $\pm$ 0.12	22.48 GiB

D Experimental setup and reproducibility

This appendix provides complete details of the network generation procedures and parameter sampling used in the experiments, enabling full reproducibility.

We first discuss fixed fan-in networks.

Graph parameters. Each network is defined by the number of root variables r, leaf variables s, and edges t, with $t = 4s$ so that every leaf has exactly four parents.

Determinism. The Python `random` module and NumPy random number generator are both initialized to a fixed seed of 1, ensuring that every generated network is exactly reproducible and consistent across all experiments.

Parent assignment. For each leaf Y_j, exactly four distinct roots are sampled uniformly at random without replacement from the r roots, ensuring fixed fan-in and preventing duplicate edges.

Isolated root repair. After sampling, any root with no children is connected by rewiring an existing edge: a leaf Y_j is chosen uniformly at random, and one of its parents that is connected to at least one other leaf is chosen uniformly at random and that parent is replaced by the isolated root; this ensures every root has at least one child while preserving both the total edge count and the fan-in of every leaf.

Leak modeling. Each leaf has one additional leak parent modeled as an independent root node, with its leak parameter fixed to 0, so that it does not contribute to the noisy-OR activation.

Parameter sampling. Root prior probabilities $P(X_i)$ and inhibitor probabilities q_i^j are each sampled independently across all roots and edges by selecting an integer uniformly at random from $\{1, \ldots, 9999\}$ and dividing by 10,000, ensuring values lie strictly in $(0, 1)$.

Edge count convention. The reported edge count t includes only BN2O edges between roots and leaves, while leak edges are excluded from t but included in the constructed circuits.

We now turn to QMR–DT–like networks.

Table 12: Controlled findings experiment on a representative QMR–DT–like BN2O (approximately 44K edges, $|Q| = 40$). Runtime is reported as mean $\pm$ 95% confidence interval over randomly sampled queries for each overlap index.

$\omega(Q)$	Inference time mean $\pm$95% CI (s)
0	0.210 ± 0.001
1	0.410 ± 0.002
2	0.838 ± 0.005
3	1.72 ± 0.01
4	3.53 ± 0.04
5	7.25 ± 0.01
6	14.5 ± 0.1
7	28.1 ± 0.1
8	56.1 ± 0.1
9	116.0 ± 0.2
10	231.9 ± 0.3

Table 13: Compilation-size statistics for baseline methods on QMR–DT–like BN2Os. We report mean compiled circuit edges with 95% CIs over 20 runs per network size. Entries without CIs correspond to deterministic outcomes.

BN2O edges	Circuit edges (mean $\pm$ 95% CI) ACE	rank-one
180	15.2K $\pm$ 0.01K	6.57K $\pm$ 0.01K
564	378K $\pm$ 13K	302K $\pm$ 3.2K
998	18.6M $\pm$ 0.06M	18M $\pm$ 0.17M
1.20K	73.7M $\pm$ 1.0M	74.6M $\pm$ 1.3M
1.26K	139M $\pm$ 1.4M	140M $\pm$ 1.4M
1.33K	Compilation failure	Compilation failure

Graph parameters. Each network is defined by scaling a base two-layer structure with 570 roots, 4,075 leaves, and 45,000 edges; for a scale parameter γ, we set $r = \lfloor 570\gamma \rfloor$, $s = \lfloor 4075\gamma \rfloor$, and $t = \lfloor 45000\gamma \rfloor$.

Determinism. The Python `random` module and NumPy random number generator are both initialized to a fixed seed of 1, ensuring that every generated network is exactly reproducible and consistent across all experiments.

Degree distributions. Leaf in-degrees for each Y_j are sampled from a truncated Zipf distribution with exponent α_Y (for leaves) set to 2.0 over the integers from 1 to $\max\{10, \lfloor 250\gamma \rfloor\}$, and root out-degrees for each X_i are sampled independently from a truncated Zipf distribution with exponent α_X (for roots) set to 2.2 over the integers from 1 to $\max\{10, \lfloor 500\gamma \rfloor\}$.

Degree sequence correction. The sampled degree sequences are adjusted by uniformly incrementing or decrementing uniformly selected entries until the total number of stubs matches exactly t, ensuring consistency with the target edge count.

Parent assignment. Edges are generated via stub matching by expanding roots and leaves according to their degrees, shuffling both stub lists independently, and pairing them element-wise, which preserves the prescribed degree sequences.

Duplicate handling. Duplicate edges are removed after pairing, and the stub-matching procedure is repeated until the number of unique edges equals t.

Isolated root repair. After construction, any root with no children is connected by rewiring an existing edge: a leaf Y_j is chosen uniformly at random, and one of its parents that is connected to at least one other leaf is chosen uniformly at random and that parent is replaced by the isolated root; this ensures every root has at least one child while preserving both the total edge count and all leaf in-degrees. This post-processing step affects only isolated roots and preserves the sampled leaf in-degree distribution and total edge density, so the resulting network remains faithful to the intended QMR–DT–like structure.

Table 14: QMR–DT–like BN2O total compilation and inference times (s), mean ± 95% CI (20 runs), $\omega(Q) \in \{0, 4, 8\}$.

BN2O edges	$\omega(Q)$	ACE	rank-one	NOICE
564	0	0.278 ± 0.003	0.138 ± 0.001	0.00936 ± 0.00008
	4	0.278 ± 0.003	0.137 ± 0.001	0.0489 ± 0.0002
	8	0.278 ± 0.003	0.138 ± 0.001	0.675 ± 0.001
998	0	1.98 ± 0.04	1.74 ± 0.03	0.0132 ± 0.0001
	4	1.99 ± 0.04	1.75 ± 0.03	0.0821 ± 0.0002
	8	1.99 ± 0.04	1.75 ± 0.03	1.19 ± 0.01
1.20K	0	Inference failure	Inference failure	0.0151 ± 0.0001
	4	Inference failure	Inference failure	0.0971 ± 0.0003
	8	Inference failure	Inference failure	1.42 ± 0.01
1.26K	0	Inference failure	Inference failure	0.0156 ± 0.0001
	4	Inference failure	Inference failure	0.101 ± 0.001
	8	Inference failure	Inference failure	1.47 ± 0.01
1.33K	0	Compilation failure	Compilation failure	0.0278 ± 0.0002
	4	Compilation failure	Compilation failure	0.119 ± 0.001
	8	Compilation failure	Compilation failure	1.57 ± 0.01
13.5M	0	Compilation failure	Compilation failure	1730 ± 270

Table 15: Uncontrolled-overlap queries for a representative 44K-edge QMR–DT–like BN2O. For each query size $|Q|$, we sample 100 random queries without constraining $\omega(Q)$. Reported statistics summarize highly skewed runtime distributions. The relative tail is defined as $100 \times (\text{90th pct.} - \text{median})/\text{median}$, where the median denotes the typical runtime.

Findings $\lvert Q\rvert$	Median (s)	IQR (s)	90th pct. (s)	Tail relative to median (%)
1	0.21	¡ 0.01	0.22	0.8
2	0.21	¡ 0.01	0.42	99
3	0.22	0.21	0.44	98
4	0.44	0.66	2.00	350
5	0.46	0.54	1.99	334
6	1.79	2.82	7.25	305

Leak modeling. Each leaf has one additional leak parent modeled as an independent root node. Unlike the fixed fan-in setting, leak parameters are active and sampled independently for each leaf by drawing an exponent uniformly at random from the interval $[-5, -2]$ and setting the leak probability to 10 raised to that value. These leak parameters are incorporated directly into the noisy-OR potentials as the baseline activation probability in the absence of parent causes.

Parameter sampling. Root prior probabilities $P(X_i)$ are generated by sampling an exponent uniformly at random from the interval $[-6, -2]$ and setting the probability to 10 raised to that value.

Inhibitor distribution. For each edge $X_i \to Y_j$, the inhibitor probability q_i^j is sampled independently across all edges from a three-component mixture: with probability 0.70 from $[0.01, 0.05]$, with probability 0.25 from $[0.05, 0.2]$, and with probability 0.05 from $[0.2, 0.8]$.

Edge count convention. The reported edge count t includes only BN2O edges between roots and leaves, while leak edges are excluded from t but included in the constructed circuits.

Table 16: Controlled-overlap queries for a representative 44K-edge QMR–DT–like BN2O with fixed overlap $\omega(Q) = 10$. Reported statistics summarize runtime variability as a function of evidence size $|Q|$, showing near-constant runtime under fixed overlap. Relative IQR is defined as $100 \times \mathrm{IQR}/\mathrm{median}$.

Findings $\lvert Q\rvert$	Median (s)	IQR (s)	Relative IQR (%)
10	219.0	5.97	2.7
12	228.4	1.94	0.9
14	228.7	1.40	0.6
16	228.5	1.94	0.9
18	229.0	2.11	0.9

Table 17: Linear regression results for $\log_2(t(Q)/c(N))$ versus $\omega(Q)$ on representative QMR–DT–like BN2Os. Slope estimates $\hat{\beta}$ are reported with 95% CIs. Fits exhibit near-perfect linearity, with $R^2 \geq 0.9999$ across all regimes.

BN2O edges	Runs	$\omega(Q)$ range	$\hat{\beta}$ (95% CI)	R^2
4.09K	240	[0, 15]	0.9930 ± 0.0004	0.99999
44K	180	[0, 11]	1.0068 ± 0.0006	0.99998
444K	90	[0, 8]	0.9985 ± 0.0009	0.99998
All networks	510	[0, 15]	0.9971 ± 0.0006	0.99995

Table 18: Prediction accuracy of $\hat{t}(Q) = c(N)2^{\omega(Q)}$ on QMR–DT–like BN2Os. Errors are reported as relative error $100 \times |t(Q) - \hat{t}(Q)|/t(Q)$.

BN2O edges	Runs	Median rel. error (%)	IQR rel. error (%)	Max rel. error (%)
4.09K	240	4.80	1.74	7.30
44K	180	3.34	2.33	7.25
444K	90	1.26	1.56	3.72
All networks	510	3.82	2.96	7.30