Proceedings of Machine Learning Research 343:711–724, 2026 Probabilistic Graphical Models (PGM)

Bayesian Network MPE via LLM-Compiled Semantic Maximizing Circuits

Victor H. S. S. Ragazzi VICTOR.RAGAZZI@UFV.BR
Department of Electrical Engineering
Federal University of Viçosa
Viçosa, Brazil

Jhonatan S. Oliveira DESOUZJH@UREGINA.CA
Department of Computer Science
University of Regina
Regina, Canada

Cory Butz CORY.BUTZ@UREGINA.CA
Department of Computer Science
University of Regina
Regina, Canada

Leonardo Bonato Felix LEOBONATO@UFV.BR
Department of Electrical Engineering
Federal University of Viçosa
Viçosa, Brazil

Editors: Gustau Camps-Valls, Manuele Leonelli and Gherardo Varando

Abstract

Recent research has shown that Large Language Models (LLMs) are able to construct the Directed Acyclic Graph (DAG) of a Bayesian Network (BN) at an accuracy level similar to the data-driven methods. A natural question that arises is whether LLMs can perform *logical inference* over these assembled graphical structures without having access to associated Conditional Probability Tables (CPTs). We tackle this question by focusing on Most Probable Explanation (MPE) inference, a task that closely aligns with LLM reasoning. We introduce a novel CPT-free *sequential semantic inference* method which is structured as a two-phase framework. An offline *semantic compilation* phase constructs a reusable *semantic maximizing circuit* by querying the LLM over assignment contexts induced by the DAG and a variable elimination ordering. An online *evaluation* phase applies evidence through circuit labels and resolves MPE queries by traversal, without making calls to the LLM at runtime. An empirical study shows that MPE accuracy surpasses 80% across six BNs and is even close to the levels reported in the literature for DAG learning, thereby demonstrating the potential of LLMs to be used as logical, parameter-free inference engines.

Keywords: Bayesian Networks, Most Probable Explanation, Large Language Models, Knowledge Compilation, Maximizing Circuits

© 2026 V.H. S. S. Ragazzi, J. S. Oliveira, C. Butz & L. Bonato Felix.

1. Introduction

A Bayesian Network (BN) is a graphical model for reasoning under uncertainty, defined by a Directed Acyclic Graph (DAG) over random variables together with a set of Conditional Probability Tables (CPTs) (Pearl, 1988; Koller and Friedman, 2009; Darwiche, 2009). Recent work has shown that Large Language Models (LLMs) can recover the DAG of a BN from variable names and metadata alone through pairwise causal querying (Kıcıman et al., 2023) or by prompting the model to produce a complete graph in a single call (Zhang et al., 2026) at accuracy levels comparable to those of data-driven structure-learning methods (Vashishtha et al., 2023; Ban et al., 2025). This raises a natural follow-up question: can LLMs answer inference queries over an assembled DAG without access to the corresponding CPTs? Among the standard inference tasks, Most Probable Explanation (MPE) is the one most semantically aligned with what LLMs can do. MPE asks only for the most probable joint assignment to the non-evidence variables, a question that can often be resolved through domain knowledge and qualitative reasoning about how variables influence one another (Chi et al., 2024), without requiring probabilities as needed for answering marginal and posterior queries.

We answer MPE queries with a two-phase framework involving *semantic compilation* followed by *online evaluation*. The central concept is that of a *semantic maximizing circuit*, which mirrors the classical maximizing circuit obtained by compiling a BN (Darwiche, 2002), but with LLM prediction at each decision node replacing numerical max over CPT entries. In the offline compilation phase, we run a modification of the Variable Elimination (VE) algorithm (Dechter, 1996; Zhang and Poole, 1994). Each step forms VE buckets, enumerates the corresponding assignment contexts induced by the bucket scopes, queries the LLM for semantically maximizing choices over those contexts, and stores the resulting decisions in a compiled representation, which we call a semantic maximizing circuit. In the online evaluation phase, given a concrete evidence configuration, evidence is applied through the circuit labels and MPE inference reduces to a lightweight traversal of the compiled semantic maximizing circuit, without issuing any LLM calls. The result is a parameter-free inference engine whose online cost is independent of the LLM. To bridge the gap between the LLM's broad qualitative priors and the specific distributions encoded in real networks, we further introduce a *context engineering* protocol in which domain experts annotate variable interactions in brief natural language and append them to the prompt during compilation. Across our benchmark BNs, this combination surpasses 80% MPE accuracy and approaches the levels reported in the literature for LLM-based DAG learning, suggesting that LLMs can serve as reliable, parameter-free MPE inference engines when the elicitation cost shifts from numerical CPTs to qualitative context.

The remainder of this paper is organized as follows. Section 2 reviews background knowledge. Section 3 presents the proposed CPT-free framework for MPE inference with LLMs, including the semantic maximizing circuit representation and the corresponding inference procedure. Section 4 reports our empirical findings. Conclusions and future work are given in Section 5.

2. Background

We briefly review BNs, MPE, maximizing circuits, and LLM-based DAG construction.

2.1. Bayesian Networks

BNs are directed graphical models that compactly represent joint probability distributions through factorization over a DAG (Pearl, 1988; Koller and Friedman, 2009; Dechter, 1996).

Definition 1 (Bayesian network) *A BN is a pair $B = (G, P)$, where $G = (\mathbf{X}, \mathbf{E})$ is a DAG over random variables $\mathbf{X} = \{X_1, \ldots, X_n\}$, each taking values in a finite domain $\mathcal{D}_i$, and P is a set of CPTs. The joint distribution factorizes as*

$$P(\mathbf{X}) = \prod_{i=1}^{n} P(X_i \mid \mathrm{pa}(X_i)),$$

where $\mathrm{pa}(X_i)$ denotes the parents of X_i in G.

Each CPT specifies $P(X_i \mid \mathrm{pa}(X_i))$ for every combination of parental values. Eliciting these numerical parameters in practice requires either large datasets or domain expert consultation.

2.2. Most Probable Explanation

Given a BN B and observed values $\mathbf{e}$ for a subset of evidence variables $E \subset \mathbf{X}$, the MPE task seeks the joint assignment to all remaining variables that is most probable given the evidence (Pearl, 1988; Koller and Friedman, 2009; Dechter, 2019).

Definition 2 (Most Probable Explanation) *The MPE problem is to find the most probable explanation x^*, defined as the assignment to all non-evidence variables that maximizes the posterior probability:*

$$\mathbf{x}^* = \arg\max_{\mathbf{x} \setminus \mathbf{e}} P(\mathbf{X} = \mathbf{x} \mid E = \mathbf{e}).$$

Exact MPE solutions are computed using VE, following the bucket-elimination framework of Dechter (1996).

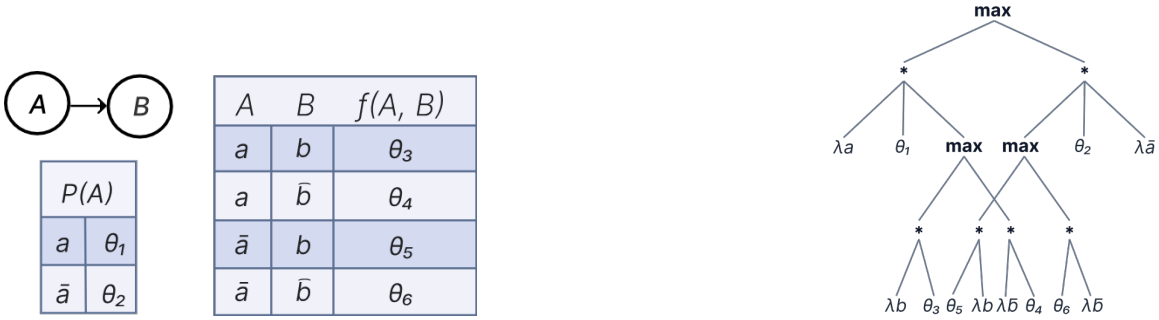
2.3. Maximizing Circuits

Maximizing circuits are a knowledge-compilation representation that encodes MPE inference in a directed acyclic circuit structure (Darwiche, 2002, 2009). A maximizing circuit is an arithmetic-circuit variant in which summation is replaced by maximization, so that evidence is applied through indicator labels and circuit evaluation computes the MPE value. As illustrated in Figure 1, the simple BN in (a) yields a maximizing circuit in (b) where leaf nodes hold CPT entries, internal nodes compute maxima over their children, and the root outputs the probability of the MPE.

2.4. LLMs for BN DAG Construction

Recent work has demonstrated that LLMs can reconstruct DAG structures from variable names and metadata alone, without access to any observational data (Vashishtha et al., 2023; Kıcıman et al., 2023; Babakov et al., 2025; Zhang et al., 2026; Ban et al., 2025). By

leveraging the domain knowledge encoded in their training corpora, these models are able to infer plausible causal orderings and edge relationships among variables, achieving accuracy levels comparable to those of data-driven structure learning methods. However, the output of such approaches is limited to the graph G: the CPTs are not produced, leaving the network incomplete for any inference task that requires assessments.



(a) A simple BN.

(b) Maximizing circuit.

Figure 1: A BN (a) and a corresponding maximizing circuit (b).

3. Semantic Compilation and Online Inference

We propose a CPT-free method for MPE inference over a BN DAG by replacing numerical CPT-based maximization with semantic branch selection performed by an LLM. The method follows the same compile-then-evaluate decomposition used by arithmetic and maximizing circuits: an offline semantic compilation phase constructs a reusable circuit representation from the DAG, and an online inference phase answers evidence-specific MPE queries by evaluating the compiled circuit. The key difference from classical maximizing circuits is the source of local decisions. In a traditional maximizing circuit, branch selection is determined by numerical CPT entries; in our method, branch selection is determined by LLM predictions over the variables, domains, and graph structure. We call the resulting representation a *semantic maximizing circuit*.

We develop the method around this compile-then-evaluate structure. We first define semantic maximizing circuits as CPT-free analogues of classical maximizing circuits. We then describe the offline semantic compilation procedure, which enumerates assignment contexts induced by the DAG and stores LLM-generated maximizing decisions. Next, we describe online inference, where a concrete evidence assignment is applied through circuit labels and the MPE prediction is obtained by traversal. Finally, we describe the LLM query design used during compilation, including qualitative annotations that can be supplied when the DAG alone is insufficient to align the LLM's semantic priors with the target network.

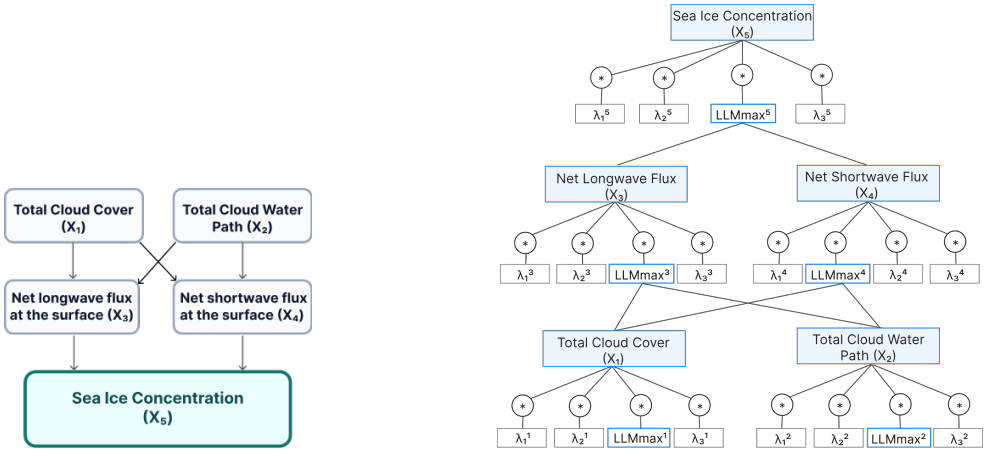
A semantic maximizing circuit formalizes the replacement of numerical CPT comparisons by LLM-based semantic decisions. Let G be a DAG over variables $\mathbf{X} = \{X_1, \ldots, X_n\}$, where each variable X_i takes values in a finite domain $\mathcal{D}_i$. In a classical maximizing circuit compiled from a BN, each max node selects the branch that maximizes a numerical

quantity derived from CPT entries. In a semantic maximizing circuit, the same circuit-level role is preserved, but the local maximizing choice is produced by an LLM from the non-numerical information available in the DAG: the target variable, its domain, its local graphical context, and the assignments appearing in the current compilation context.

Definition 3 (Semantic maximizing circuit) *Let G be a DAG over variables $\mathbf{X}$ with finite domains $\mathcal{D}_1, \ldots, \mathcal{D}_n$. A* semantic maximizing circuit *for G is a directed acyclic circuit whose structure mirrors a maximizing circuit over the variables in G, but whose local maximizing decisions are generated without CPT entries. Each decision node associated with a variable X_i stores an LLM-selected value $\hat{x}_i \in \mathcal{D}_i$ for each relevant assignment context considered during semantic compilation. Evidence assignments are applied at online evaluation time through the corresponding circuit labels.*

Example 1 illustrates how a semantic maximizing circuit encodes the dependency structure of a sub-DAG and handles evidence queries through indicator activation. In the next subsection, we show how such circuits are compiled from a Bayesian network and an elimination order.

Example 1 *In Figure 2(a), the variables* Total Cloud Cover *(X_1) and* Total Cloud Water Path *(X_2) influence* Net Longwave Flux *(X_3) and* Net Shortwave Flux *(X_4), which in turn influence* Sea Ice Concentration *(X_5), forming a sub-DAG of the full Bayesian network (Huang et al., 2021). The corresponding semantic maximizing circuit in Figure 2(b) preserves the maximizing-circuit view of this dependency structure. Each variable has indicator labels for its possible values, and the circuit contains max nodes whose selected branches encode semantic maximizing choices. An evidence query such as X_5=High is handled by activating the corresponding X_5 label during online evaluation.*



(a) Example DAG subgraph.

(b) Semantic maximizing circuit.

Figure 2: Figure (a) shows a manually selected subgraph of the full Arctic Sea Ice network (Huang et al., 2021). Figure (b) shows the corresponding semantic maximizing circuit.

3.1. Offline Compilation

We compile the circuit as an offline representation by following VE elimination order to construct a sequence of LLM calls. A query-time realization of semantic MPE would call the LLM each time an MPE query is asked, repeatedly solving similar local decision problems and incurring latency and monetary cost at runtime. Compilation avoids this by generating and storing the semantic decisions in advance. Once the semantic maximizing circuit has been compiled, subsequent evidence-specific queries can be answered by circuit evaluation and lookup, without calling the LLM again.

Offline compilation follows a VE style procedure over the DAG, as summarized in Algorithm 1. Variables are processed according to a min-degree elimination ordering π, and for each variable the compiler enumerates the assignment contexts induced by its bucket scope. The LLM is queried once per variable, returning one maximizing decision per context row, which is then stored in the corresponding max node of the semantic maximizing circuit.

Algorithm 1: Offline Semantic Compilation

Input: DAG G over $\mathbf{X} = \{X_1, \ldots, X_n\}$ with domains $\{\mathcal{D}_i\}$; elimination order π; LLM oracle $\mathcal{M}$

Output: Semantic maximizing circuit $\mathcal{C}$

1 Initialize $\mathcal{C}$ with one indicator leaf node (X_i, d) for each variable $X_i \in \mathbf{X}$ and each value $d \in \mathcal{D}_i$;
2 Assign each factor in G to the bucket of its highest-priority variable under π;
3 **for** *each variable* X_i *in order* π **do**
4 Let $\mathbf{S}_i \leftarrow \text{scope}(B_i) \setminus \{X_i\}$; `// remaining variables in bucket` B_i
5 Let $\mathcal{R}_i \leftarrow \prod_{X_j \in \mathbf{S}_i} \mathcal{D}_j$; `// Cartesian product of domains of variables in` $\mathbf{S}_i$
6 Query $\mathcal{M}$ with the variable name X_i, its domain $\mathcal{D}_i$, and the context table $\mathcal{R}_i$;
7 Receive decisions $\hat{\mathbf{x}}_i = \{\hat{x}_i(\mathbf{r})\}_{\mathbf{r} \in \mathcal{R}_i}$, one value per row;
8 **for** *each* $\mathbf{r} \in \mathcal{R}_i$ **do**
9 Let $v \leftarrow \hat{x}_i(\mathbf{r})$;
10 Store the pair $(\mathbf{r}, v)$ as an entry in the max node of $\mathcal{C}$ for X_i;
11 **end**
12 Remove X_i from G;
13 Create a new factor over $\mathbf{S}_i$;
14 Assign this factor to the bucket of the highest-priority variable in $\mathbf{S}_i$ under π;
15 **end**
16 **return** $\mathcal{C}$;

Example 2 *Consider the sub-DAG in Figure 2(a) over variables* $\{X_1, X_2, X_3, X_4, X_5\}$*, and suppose the reverse topological elimination order is* $\pi = (X_5, X_4, X_3, X_2, X_1)$*, so* X_5 *is processed first.*

Scope identification (Algorithm 1, line 4). *The compiler collects the scope of bucket* B_{X_5} *and removes* X_5 *itself, yielding* $\mathbf{S}_{X_5} = \{X_3, X_4\}$*.*

Context enumeration (line 5). *The context rows are formed as the Cartesian product of the domains of all variables in* $\mathbf{S}_{X_5}$*. Assuming* $\mathcal{D}_{X_3} = \mathcal{D}_{X_4} = \{Low, Medium, High\}$*, this*

yields $|\mathcal{R}_{X_5}| = 3 \times 3 = 9$ *rows:*

$$\mathcal{R}_{X_5} \;=\; \{\mathit{Low},\, \mathit{Medium},\, \mathit{High}\} \times \{\mathit{Low},\, \mathit{Medium},\, \mathit{High}\}.$$

LLM query (lines 6–7). *The oracle* $\mathcal{M}$ *receives the variable name* X_5 *(Sea Ice Concentration), its domain* $\mathcal{D}_{X_5} = \{\mathit{Low}, \mathit{Medium}, \mathit{High}\}$*, and the full context table* $\mathcal{R}_{X_5}$*, whose rows are all combinations of Net Longwave Flux (*X_3*) and Net Shortwave Flux (*X_4*) values. It returns one estimated maximizing assignment per row:*

$$\hat{\mathbf{x}}_{X_5} = \begin{cases} \mathit{High} & \text{if } X_3 = \mathit{Low},\ X_4 = \mathit{Low}, \\ \mathit{High} & \text{if } X_3 = \mathit{Low},\ X_4 = \mathit{Medium}, \\ \quad\vdots & \\ \mathit{Low} & \text{if } X_3 = \mathit{High},\ X_4 = \mathit{High}. \end{cases}$$

Decision storage (lines 8–11). *Each row* $\mathbf{r} \in \mathcal{R}_{X_5}$ *is paired with the returned decision and stored in the max node for* X_5*. For instance, the first row yields the entry:*

$$\big(X_3 = \mathit{Low},\ X_4 = \mathit{Low}\big) \;\mapsto\; \hat{x}_{X_5} = \mathit{High}.$$

All nine pairs are stored analogously. During online inference, once values for X_3 *and* X_4 *are available, the corresponding entry is retrieved directly by lookup, without any further LLM call.*

Elimination and propagation (lines 12–14). X_5 *is removed from* G *(line 12), a new factor over* $\mathbf{S}_{X_5} = \{X_3, X_4\}$ *is created (line 13), and assigned to the bucket of the variable in this scope that appears earliest in the remaining elimination order (line 14), namely* X_4*.*

This illustrates the full iteration for a single variable. The same procedure is applied to each remaining variable in order (X_4, X_3, X_2, X_1)*, ultimately producing the complete semantic maximizing circuit* $\mathcal{C}$*.*

Example 2 shows the local form of offline compilation: one eliminated variable induces a table of assignment contexts, and the LLM supplies one semantic maximizing decision per row. Repeating this procedure across the elimination order yields the full compiled circuit used by the online phase.

3.2. Online Inference with Evidence

Online inference applies a concrete evidence assignment to the compiled semantic maximizing circuit. Given evidence $\mathbf{e}$, the indicator leaves consistent with $\mathbf{e}$ are activated and the remaining ones are turned off. The circuit is then traversed in topological order over the non-evidence variables: starting from $\hat{\mathbf{x}} = \emptyset$, each variable X_i is visited in turn, and its stored decision is retrieved by a table lookup keyed on the current context $\mathbf{S}_i$. This context is formed by combining the evidence values of any parent fixed by $\mathbf{e}$ with the decisions already accumulated in $\hat{\mathbf{x}}$ for the remaining parents, which are guaranteed to have been resolved earlier in topological order. The retrieved value $\hat{x}_i$ is then appended to $\hat{\mathbf{x}}$, and the process repeats until all non-evidence variables are assigned. The final assignment is returned as the predicted MPE configuration.

Since all decisions were generated during offline compilation, online inference requires no LLM calls. Its cost is determined solely by circuit traversal and table lookup rather than by LLM latency.

Example 3
We continue from Example 2, using the compiled circuit $\mathcal{C}$ over $\{X_1, X_2, X_3, X_4, X_5\}$. Online reconstruction follows the reverse of the compilation order.

The online query sets $\mathbf{e} = \{X_1 = High\}$ (Total Cloud Cover). Evaluation activates the indicator leaf $(X_1, High)$ and deactivates (X_1, Low) and $(X_1, Medium)$. Since X_1 is fixed by evidence, traversal proceeds over the non-evidence variables in topological order (X_2, X_3, X_4, X_5), with $\hat{\mathbf{x}} = \emptyset$.

***Step 1, resolving** X_2. Although X_2 (Total Cloud Water Path) has no parents in the original DAG, its compiled max node has context $\mathbf{S}_{X_2} = \{X_1\}$ due to factors propagated during elimination. With $X_1 = High$, the lookup returns $\hat{\mathbf{x}} = \{X_2 = Medium\}$.*

***Step 2, resolving** X_3. The max node for X_3 (Net Longwave Flux) was compiled with context $\mathbf{S}_{X_3} = \{X_1, X_2\}$. With $X_1 = High$ fixed by evidence and $X_2 = Medium$ already assigned, the lookup key is $(X_1 = High, X_2 = Medium)$, $X_3 = Low$, giving*

$$\hat{\mathbf{x}} = \{X_2 = Medium,\ X_3 = Low\}.$$

The same lookup procedure is applied to each remaining variable (X_4, X_5) in turn, yielding the complete MPE assignment

$$\hat{\mathbf{x}} = \{X_2 = Medium,\ X_3 = Low,\ X_4 = Low,\ X_5 = High\}$$

without any call to LLM at runtime.

The next section describes how this method is evaluated empirically, beginning with the query design used during compilation.

4. Experiments

We empirically evaluate semantic maximizing circuits for MPE inference across six benchmark Bayesian networks from the `bnlearn` repository (Scutari, 2010), spanning finance, cellular biology, forensics, infectious disease, and risk modeling. The networks range from 7 to 27 nodes and from treewidth 1 to 8. Evidence is introduced incrementally, from a single observed node up to 50% of the network. We compare circuit assignments against the exact MPE solution computed by VE, reporting variable-level accuracy and weighted F1. All experiments use a single fixed LLM with deterministic decoding and the same min-degree elimination ordering.

All experiments use GPT-5.4 with temperature 0.0 and are implemented in Python using the `pgmpy` library (Ankan and Textor, 2024). Evidence variables are selected at random, while all remaining experimental conditions are held fixed across runs.

Recall the usage of an LLM M in line 6 of Algorithm 1 for deciding on assignments. The query process itself can be designed along two orthogonal dimensions. The *prompt template* is the fixed syntactic structure shared by every LLM call during compilation: the ordering and labeling of fields, the instruction format, and the output schema that enforces one

legal-state decision per context row. The *prompt content* is the variable semantic payload instantiated for each call: the target variable, its domain, the relevant assignment context rows, local family information, separator variables, and any qualitative annotations available for the network.

The previous subsections described where the stored decisions appear in the compiled circuit and how they are used during online inference. Here we describe how those decisions are generated, following this template–content decomposition.

Prompt template. The template serves both a structural and an inferential purpose. Structurally, it presents the local graphical information in a consistent format and requires a machine-readable response, so that every compilation call yields valid entries for the semantic maximizing circuit. Inferentially, it guides the LLM to make a local MPE-style semantic choice without inventing probabilities or variables. This preserves the CPT-free nature of the method while making the LLM act as a parameter-free expert for qualitative maximizing decisions.

Prompt content. Qualitative context annotations can be added when the DAG topology and variable names alone do not sufficiently characterize the target distribution. These annotations describe reusable local mechanisms, such as whether one biological pathway suppresses another or whether a risk factor is rare in a given population. They do not provide numerical CPT entries. Instead, they supply short natural-language descriptions of variable interactions that help align the LLM's semantic priors with the specific network being compiled. In this sense, the elicitation burden shifts from numerical parameter specification to qualitative mechanism description.

Scalability. Since each variable is queried only once, regardless of the size of $|\mathcal{R}_i|$, the compilation process requires exactly n LLM requests per network. Importantly, the bucket scope $\mathbf{S}_i$ and context table $\mathcal{R}_i$ for each variable are determined solely by the DAG and the elimination order, and do not depend on the semantic decisions made for other variables. This means that the n requests are structurally independent and can be processed in parallel. In practice, we batch these requests according to the LLM provider's requests-per-minute (RPM) limit, allowing compilation to be completed in $\lceil n/\mathrm{RPM} \rceil$ batches.

Example 4 *For a biological signaling network, an annotation may state:*

> *Do not assume Akt copies ERK or PKA. Akt is a survival and growth kinase that receives permissive pressure from MAPK signaling while also being shaped by PKA cross-talk. At a fixed ERK level, stronger PKA modulation can suppress Akt below baseline, while lower PKA can leave Akt at AVG. HIGH Akt needs a coherent survival-signaling push; LOW Akt can reflect dampened transduction rather than LOW ERK.*

Such an annotation is appended to the relevant compilation prompts and affects the semantic decisions stored in the circuit. Once compilation is complete, the same online evaluation procedure applies: evidence is handled through circuit labels and lookup traversal, with no additional LLM calls.

Example 4 illustrates how qualitative context annotations refine the decisions produced during semantic compilation without changing the online inference procedure.

4.1. Prompt Template Comparison

We first assess how well an LLM can perform MPE inference using only the graph structure and variable names, isolating the contribution of the prompt template itself. We compare three templates with increasing levels of reasoning: the *simple* template requests a direct answer, *CoT* additionally asks the model to reason before deciding, and *guided CoT* provides explicit step-by-step guidance, including checking parent states before making a decision.

Table 1 reports mean accuracy across the six networks. Structured reasoning provides modest but consistent improvements: guided CoT outperforms simple prompting by +8.3 pp on average, indicating that reasoning instructions are beneficial even without domain-specific information. Nevertheless, the best template reaches only 57.1% accuracy, showing that prompt design alone is insufficient for reliable MPE inference.

Table 1: Mean accuracy across benchmark networks with no qualitative annotations, across three prompt template variants.

Dataset	Simple	CoT	Guided CoT
All 6 datasets	48.8%	55.8%	57.1%

4.2. Effect of Evidence Without Qualitative Annotations

A natural expectation is that more observed evidence should help the LLM narrow down the most probable configuration. We test this by introducing evidence incrementally—from a single observed node up to 50% of the network—while keeping the template-only content configuration and the guided CoT template fixed.

Table 2 reports accuracy at single evidence and at the maximum evidence ratio for each network. The results contradict the expectation: performance is neither consistent nor monotone. Most networks degrade as evidence grows, with wrong early decisions compounding through the lookup chain, while a few oscillate without a clear trend. The overall picture is one of unreliable inference—more evidence does not help when the LLM's priors are misaligned with the target network, and in several cases it actively hurts.

Table 2: Direct LLM inference accuracy under single evidence and maximum evidence ratio. Structural properties include treewidth (TW) and average Markov blanket size (AvgMB).

Dataset	Nodes	TW	AvgMB	Domain	Acc (1 ev.)	Acc (max ev.)
Insurance	27	6	5.19	Insurance	34.6%	40.5%
Sachs	11	3	3.09	Biology	36.7%	27.8%
Coronary Risk	9	8	8.00	Medicine	58.3%	40.0%
Crimescene	9	4	4.22	Forensics	58.3%	60.0%
Gonorrhoeae	10	3	3.20	Medicine	74.1%	60.0%
Cryptocurrency	7	1	1.43	Finance	88.9%	41.7%

4.3. Adding Qualitative Content

The results above suggest that the limiting factor is not the template structure but the absence of domain-specific content. We now evaluate the method with qualitative annotations included in the prompt content, while olding evidence fixed at the single-observation setting.

Qualitative annotations describing local variable interactions are included in the compilation prompts to align the LLM's decisions with the target network's mechanisms (see Example 4).

Table 3 compares prompting without qualitative annotations against prompting with qualitative annotations at single-evidence configuration. The method improves on every evaluated network, with gains ranging from no change on Cryptocurrency—already near ceiling—to +39.8 pp on Insurance, where generic LLM priors were most misaligned with the specific signaling pathway mechanisms. Beyond raw accuracy, results also become more consistent: whereas the annotation-free approach fluctuated substantially across networks, all four evaluated networks now cluster between 70% and 89%.

The single "n.a." result in Table 3 arises from a configuration in which the variable elimination order yields a large parent set for the target node. The resulting context — further expanded by the qualitative annotations — grows large enough that the LLM's response becomes unstable and fails to return a complete, well-formed maximization over the variable's domain.

Table 3: Effect of qualitative prompt content on inference accuracy at single-evidence configuration. Networks marked *n/a* were not evaluated with annotations.

	No annotations		**With annotations**	
Dataset	**Acc**	**F1-W**	**Acc**	**F1-W**
Sachs	36.7%	0.276	70.0%	0.766
Insurance	34.6%	0.362	74.4%	0.749
Coronary Risk	58.3%	0.565	*n/a*	*n/a*
Crimescene	58.3%	0.547	79.2%	0.784
Gonorrhoeae	74.1%	0.713	81.5%	0.802
Cryptocurrency	88.9%	0.838	88.9%	0.889

4.4. Effect of Evidence With Qualitative Content

We now examine whether the compiled circuit integrates additional observations coherently as evidence grows.

Figure 3 shows accuracy across evidence ratios for the four evaluated networks. The behavior contrasts sharply with the template-only regime: none of the networks degrades as evidence grows. Sachs in particular rises from 70.0% at single evidence to 93.8% at 25% observed nodes, suggesting that the compiled circuit propagates additional observations coherently rather than compounding early errors. More evidence does not guarantee monotone improvement—Gonorrhoeae oscillates modestly around its baseline—but it does not hurt, which is a meaningful behavioral difference from the results in Section 4.2.

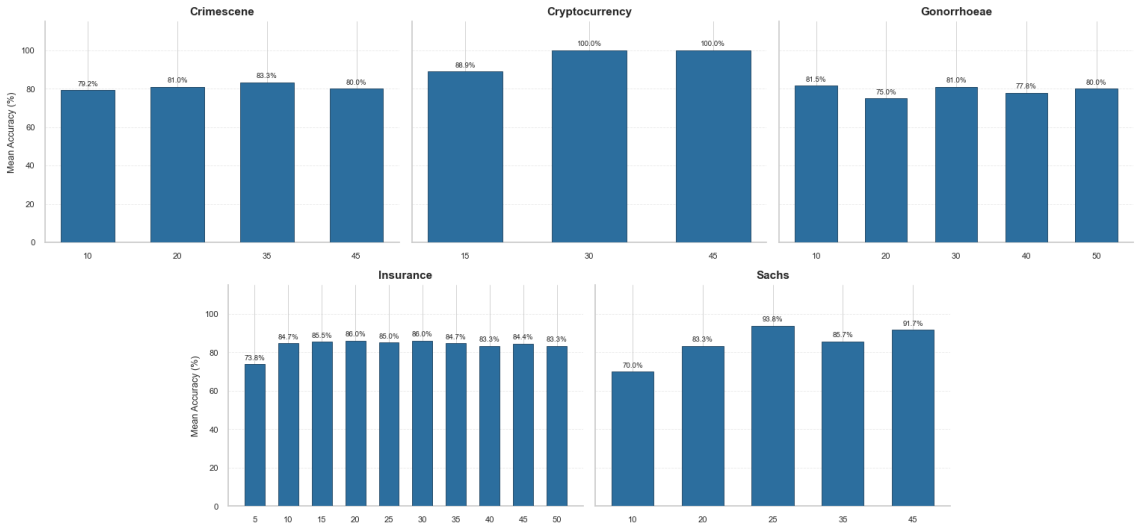


Figure 3: Mean inference accuracy as a function of evidence ratio for the semantic maximizing circuit with qualitative annotations.

5. Conclusion

This paper studied whether LLMs can support MPE inference over a BN DAG when CPTs are unavailable. We introduced the *semantic maximizing circuit*, a CPT-free analogue of a classical maximizing circuit in which numerical maximization over CPT entries is replaced by LLM-generated semantic decisions. The method separates semantic knowledge elicitation from evidence-specific inference: offline semantic compilation generates and stores decisions through a VE-style procedure, while online evaluation applies evidence through circuit labels and traverses the compiled semantic maximizing circuit without runtime LLM calls. The compilation cost is governed by the induced width of the elimination ordering and the cardinalities of the variables in each bucket scope, inheriting the same worst-case combinatorial profile as classical knowledge compilation.

The experiments show that the main bottleneck is semantic alignment rather than graph structure alone. Direct LLM inference—querying the model for the most probable state of each variable without any compilation structure—produces inconsistent results across networks and degrades as evidence grows, regardless of prompt template complexity. The compiled semantic maximizing circuit substantially improves over this baseline on every evaluated network, with gains driven by qualitative context annotations that supply brief descriptions of variable interactions during compilation, shifting the elicitation burden from numerical CPTs to qualitative domain knowledge. The method therefore offers a parameter-free route to approximate MPE inference when a DAG is available but CPT elicitation or parameter learning is impractical, while retaining the usual scalability limitations of compilation and introducing a dependence on domain annotations. Future work includes automating these annotations, selecting evidence configurations that reduce semantic conflict, approximating large context spaces, and extending the framework beyond MPE to MAP and posterior queries.

References

Ankur Ankan and Johannes Textor. pgmpy: A python toolkit for bayesian networks. *Journal of Machine Learning Research*, 25(265):1–8, 2024. URL http://jmlr.org/papers/v25/23-0487.html.

Nikolay Babakov, Ehud Reiter, and Alberto Bugarín. Scalability of Bayesian network structure elicitation with large language models: a novel methodology and comparative analysis. In *Proceedings - International Conference on Computational Linguistics, COLING*, volume Part F206484-1, pages 10685–10711. Association for Computational Linguistics (ACL), 2025. ISBN 9798891761964.

Taiyu Ban, Lyuzhou Chen, Derui Lyu, Xiangyu Wang, Qinrui Zhu, Qiang Tu, and Huanhuan Chen. Integrating large language model for improved causal discovery. *IEEE Transactions on Artificial Intelligence*, 6(11):3030–3042, 2025. doi: 10.1109/TAI.2025.3560927.

Haoang Chi, He Li, Wenjing Yang, Feng Liu, Long Lan, Xiaoguang Ren, Tongliang Liu, and Bo Han. Unveiling causal reasoning in large language models: Reality or mirage? In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 96640–96670. Curran Associates, Inc., 2024. doi: 10.52202/079017-3064. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/af2bb2b2280d36f8842e440b4e275152-Paper-Conference.pdf.

Adnan Darwiche. A logical approach to factoring belief networks. In *Proceedings of the Eighth International Conference on Principles of Knowledge Representation and Reasoning (KR'02)*, pages 409–420, 2002.

Adnan Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, New York, NY, 2009.

Rina Dechter. Bucket elimination: A unifying framework for probabilistic inference. In *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 211–219. Morgan Kaufmann, 1996.

Rina Dechter. *Reasoning with Probabilistic and Deterministic Graphical Models*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Springer, Cham, 1 edition, 2019.

Yiyi Huang, Matthäus Kleindessner, Alexey Munishkin, Debvrat Varshney, Pei Guo, and Jianwu Wang. Benchmarking of data-driven causality discovery approaches in the interactions of arctic sea ice and atmosphere. *Frontiers in Big Data*, Volume 4 - 2021, 2021. ISSN 2624-909X. doi: 10.3389/fdata.2021.642182.

Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

Emre Kıcıman, Robert Osazuwa Ness, Amit Sharma, and Chenhao Tan. Causal reasoning and large language models: Opening a new frontier for causality, 2023. URL https://api.semanticscholar.org/CorpusID:258426662.

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

Marco Scutari. Learning Bayesian networks with the bnlearn R package. *Journal of Statistical Software*, 35(3):1–22, 2010. doi: 10.18637/jss.v035.i03.

Aniket Vashishtha, Abbavaram Gowtham Reddy, Abhinav Kumar, Saketh Bachu, Vineeth N. Balasubramanian, and Amit Sharma. Causal inference using llm-guided discovery. 2023. URL https://www.micusal-inference-using-llm-guided-discovery/.

Nevin L. Zhang and David Poole. A simple approach to Bayesian network computations. In *Proceedings of the Tenth Canadian Conference on Artificial Intelligence*, pages 171–178, 1994.

Yinghuan Zhang, Yufei Zhang, Parisa Kordjamshidi, and Zijun Cui. Bayesian network structure discovery using large language models. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=G4mrO8LVix.