LEARNING PROBABILISTIC GRAPHS FROM NATURAL LANGUAGE

Anonymous authors
Paper under double-blind review

ABSTRACT

Natural-language descriptions contain knowledge about uncertain systems, but using that knowledge in probabilistic models requires an explicit account of variables, states, dependencies, and probabilities. Recovering these elements together is difficult because errors in one component can change the meaning of the others. We introduce **DUAL-BN**, a two-stage framework for learning this translation through bidirectional supervision and structured feedback. A shared language model first learns to extract probabilistic graph components from text and reconstruct descriptions from graphs. Group Relative Policy Optimization (GRPO) then refines jointly generated components using reference-graph fidelity, semantic consistency of the text–graph–text cycle, and a penalty for component counts exceeding the reference. The probability targets are parent-specific pairwise conditional tables. On PRISM-BN backbones unseen during training, DUAL-BN increases Node F1 from 0.7745 to 0.9106 and mapped Edge F1 from 0.6762 to 0.8415 over supervised fine-tuning, while reducing forward KL divergence on matched pairwise tables from 0.8707 to 0.6594. The trained policy is evaluated through staged component queries with reference-assisted filtering. The gains accompany a decrease in conditional State F1. A separate study of composition from supplied partial graphs examines how such representations can support larger probabilistic backbones. On ten descriptions rated by three evaluators, mean overall reconstruction quality rises from $3.41/5$ for SFT to $4.10/5$ for DUAL-BN. The results support combining token-level supervision with feedback on the structure and recoverable information of graph predictions as a step toward building probabilistic models from language.

1 INTRODUCTION

Natural language often describes uncertain systems without making their probabilistic structure explicit. A passage may introduce variables, describe possible outcomes, and explain how one condition changes the likelihood of another, with the relevant information spread across several sentences. Turning this knowledge into a probabilistic model requires making those variables, outcomes, and relationships explicit. Bayesian networks (BNs) provide a representation through discrete states, directed dependencies, and conditional probabilities (Koller & Friedman, 2009). Recovering their components from text connects the knowledge expressed in language with models whose assumptions can be inspected and developed for probabilistic reasoning.

The difficulty lies in coordinating these components. Variables must refer to the intended concepts, states must define their domains, edges must express the directed relationships, and probability tables must agree with the chosen variables and states. An extra variable can introduce additional states, dependencies, and parameters. A reversed edge changes the conditional being represented. A well-formed output can therefore describe a substantially different system from the source text. Prior work has studied BN structure elicitation (Babakov et al., 2025; Zhang et al., 2026) and probability estimation for supplied structures (Nafar et al., 2025). We study learning to recover variables, states, dependencies, and probability information together from paired descriptions and reference graphs. Our target uses a conditional table for each parent–child pair. These local tables describe benchmark dependencies, but do not by themselves specify a complete BN.

Supervised fine-tuning (SFT) teaches a model to reproduce serialized reference outputs, but its token loss does not directly compare the structures represented by alternative completions. A well-formatted

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.

candidate can omit a variable, reverse a dependency, or misrepresent probabilities. We therefore combine token supervision of the text–graph mappings with rewards for the predicted components and the meaning recoverable from them.

We introduce **DUAL-BN**, a two-stage framework for probabilistic graph reconstruction. In the first stage, a shared language model learns both component extraction from text and graph-to-text reconstruction. The reverse mapping gives each predicted graph a textual interpretation that can be compared with the source description. In the second stage, Group Relative Policy Optimization (GRPO) (Shao et al., 2024) refines candidates that contain nodes, states, edges, and pairwise tables in a single completion. The reward combines direct reference fidelity with semantic consistency of the text–graph–text cycle. Direct matching assesses individual components and probability entries, while reconstruction assesses the information expressed by the graph through its generated description. We also penalize component counts that exceed the reference. These counts include edges and probability parameters outside the matched structure. Reference matching, textual reconstruction, and candidate-wide counts therefore assess different properties of a prediction. We evaluate their combined design and compare refinement configurations with and without the excess penalty.

We evaluate on PRISM-BN, which contains 5,054 description–graph pairs from 50 source backbones (Bhattacharya et al., 2026). All descriptions from a backbone remain in the same partition, testing transfer to source systems unseen during training. We evaluate the trained policy through staged component queries, using reference matches to select downstream inputs (Section 5). Under this protocol, DUAL-BN increases Node F1 from 0.7745 to 0.9106 and mapped Edge F1 from 0.6762 to 0.8415 over SFT. Forward KL on matched benchmark pairwise tables decreases from 0.8707 to 0.6594. GRPO without complexity control improves all five reported means relative to SFT. The full framework achieves the strongest node, edge, and forward-KL results, with conditional State F1 decreasing from 0.8738 under SFT to 0.8486.

Individual descriptions often cover only part of a larger system. To examine the next modeling step, we also study prompted composition of supplied partial graphs into probabilistic backbones. This separate experiment tests how partial representations can be consolidated and identifies gaps between structural reconstruction, probability coverage, and faithful verbalization. Human evaluations compare SFT, both refinement configurations, and GPT-6 Astra on ten subgraph examples. The full framework receives a mean overall rating of $4.10/5$, compared with $3.41/5$ for SFT. We also assess ten backbones constructed from regularized DUAL-BN predictions using GPT-5.6.

Our contributions are:

- We introduce DUAL-BN, a two-stage framework combining bidirectional text–graph supervision with GRPO rewards for reference fidelity, semantic reconstruction, and excess component counts.
- We evaluate graph-level refinement with and without excess-count control on held-out source backbones, reporting gains over SFT in node, edge, and pairwise probability reconstruction under staged evaluation.
- We analyze reconstruction quality through component scores, human comparisons, and individual predictions, and assess composition of supplied partial graphs and DUAL-BN predictions into larger backbones.

2 Related Work

Learning probabilistic representations from language. Prior work connects natural language to explicit probabilistic models through program translation (Wong et al., 2023) and studies reasoning over descriptions that contain probabilistic information (Nafar et al., 2024; Schrader et al., 2024). For BNs, language models have been used to elicit structure (Babakov et al., 2025), refine candidate networks (Zhang et al., 2026), and coordinate structural reasoning through agents (Cai et al., 2026). Other work estimates probabilities for supplied structures (Nafar et al., 2025) or fits BNs for downstream inference (Feng et al., 2025). DUAL-BN learns to reconstruct variables, states, directed dependencies, and pairwise conditional tables from the paired descriptions and reference graphs in PRISM-BN (Bhattacharya et al., 2026). It combines bidirectional supervised fine-tuning with GRPO to learn these components together. Reference matching and graph-to-text reconstruction provide feedback on the predicted components and the information they express.

Bidirectional learning and reconstruction. Dual learning couples opposite translation directions through reinforcement learning (He et al., 2016), and dual semantic parsing uses validity and reconstruction signals for language and logical forms (Cao et al., 2019). CycleGT learns text-to-graph and graph-to-text mappings from nonparallel data (Guo et al., 2020). Related approaches use cycle training for faithful data-to-text generation (Wang et al., 2023) and learn across multiple conversion directions (Duong et al., 2023). We adapt bidirectional supervision and reconstruction feedback to probabilistic graph components, which require consistent variable identities, state spaces, edge directions, and numerical tables. The shared model reconstructs a description from each candidate graph. Comparing that description with the source provides semantic feedback alongside direct component matching. Direct matching can expose incorrect probability entries and structural errors that a fluent reconstruction may conceal.

Graph-level rewards and complexity. GRPO updates a policy by comparing rewards across sampled completions (Shao et al., 2024). DUAL-BN applies this optimizer to candidates that contain all predicted graph components. Its reward combines agreement with the reference graph and semantic agreement between the source and reconstructed descriptions, using an LLM judge for semantic comparisons (Zheng et al., 2023). Complexity penalties are established in BN learning, including minimum-description-length objectives (Lam & Bacchus, 1994). Our penalty counts nodes, edges, states, and probability parameters in excess of the reference. It uses supervised reference information and penalizes these excess counts rather than total graph size or description length. We compare GRPO with and without this penalty while retaining the other reward terms.

Combining partial graphs. BN fusion combines network structures (Matzkevich & Abramson, 1992), including local Gaussian models learned from data (Gu & Zhou, 2020). Incremental knowledge-graph construction also integrates entities and relations across inputs (Lairgi et al., 2024). Our separate composition study asks whether prompted models can consolidate supplied partial probabilistic graphs and describe the resulting backbone. The study examines how local views can be combined into larger representations. Its inputs are supplied graphs rather than DUAL-BN predictions, and model parameters remain fixed. Section 6.1 summarizes the study, with the full results in Appendix C.5.

3 Task and Data

Data and target representation. PRISM-BN contains 5,054 generated descriptions paired with reference graphs derived from 50 source backbones across five domains (Bhattacharya et al., 2026). Each backbone represents a larger system, and its subgraphs provide partial views. We use the graph-specific generated descriptions, rather than the original source articles. The probabilities are internally constructed benchmark targets. Agreement with them measures reconstruction fidelity. The descriptions convey relationship strength without transcribing every probability entry, so they may not uniquely determine the numerical targets (Bhattacharya et al., 2026).

For a description x, the extraction target is

$$G = (V, \mathcal{S}, E, \mathcal{Q}), \qquad \mathcal{S} = \{\mathcal{S}_v : v \in V\}, \quad \mathcal{Q} = \{Q_{uv} : (u, v) \in E\}. \tag{1}$$

Here V contains variables, $\mathcal{S}_v$ is the ordered state list of variable v, and E contains directed dependencies. For $u \to v$,

$$Q_{uv}[a, b] = P(v = s_{v,a} \mid u = s_{u,b}), \qquad Q_{uv} \in [0, 1]^{r_v \times r_u}, \quad \sum_a Q_{uv}[a, b] = 1, \tag{2}$$

where $r_v = |\mathcal{S}_v|$. Rows index child states and columns index parent states. We use the parent-specific matrices stored in the corpus. PRISM constructs these local conditionals by normalizing elicited pairwise joint tables (Bhattacharya et al., 2026). We do not obtain them by marginalizing full multi-parent CPDs or assume that they equal conditionals implied by a single global joint distribution. Appendix A describes target preprocessing. The corpus also provides root priors and full multi-parent conditionals. Our target omits root priors and $P(v \mid \mathrm{Pa}(v))$ for joint parent assignments. These pairwise tables alone do not determine a general BN joint distribution.

Backbone-disjoint partition. We randomly assign 40 backbones to training, five to validation, and five to testing using seed 42. All descriptions from a backbone remain in the same partition. Unequal

numbers of subgraphs per backbone produce unequal partition sizes (Table 1). Validation and test examples are excluded from parameter updates.

	Train	Validation	Test	Total
Backbones	40	5	5	50
Description–graph pairs	4,240	741	73	5,054
Pairwise-table examples	31,742	5,731	452	37,925
All instruction examples	48,702	8,695	744	58,141

Table 1: Extraction data. Each graph contributes four graph-level instruction examples and one probability example per edge. Edge occurrences are counted separately across overlapping subgraphs.

4 DUAL-BN

DUAL-BN uses a shared DeepSeek-R1-Distill-Qwen-14B model for extraction and graph-to-text generation. Training proceeds in two stages (Figure 1). Supervised initialization teaches component extraction and graph verbalization from paired examples. Reward refinement scores each joint prediction by matching its components to the reference, comparing its reconstructed description with the source, and penalizing excess component counts. This moves from token-level supervision of individual tasks to feedback on the components represented together in each candidate. Both stages adapt low-rank parameters using QLoRA (Dettmers et al., 2023).

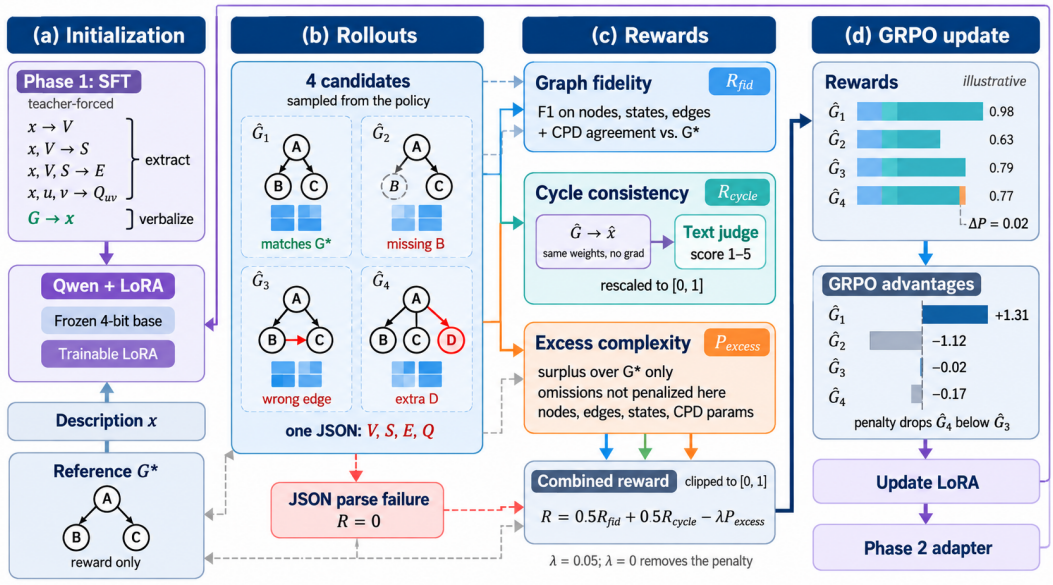


Figure 1: DUAL-BN training. Bidirectional SFT initializes the shared policy. Four candidate graphs illustrate reference recovery, an omitted node, an incorrect edge, and an extra node. Fidelity, text-cycle agreement, and excess counts determine the GRPO reward. CPD blocks represent parent-specific pairwise conditional tables. Reward values and advantages are illustrative. The update changes the shared adapter, while reverse generation supplies a scalar reward without backpropagation through verbalization.

4.1 Bidirectional Supervised Initialization

We pool five instruction types:

$$x \longrightarrow V, \qquad (x, V) \longrightarrow \mathcal{S}, \qquad (x, V, \mathcal{S}) \longrightarrow E,$$
$$(x, u, \mathcal{S}_u, v, \mathcal{S}_v) \longrightarrow Q_{uv}, \qquad \text{serialize}(G) \longrightarrow x. \tag{3}$$

The forward tasks follow the dependencies of the representation: variables identify the state spaces to describe and the possible edge endpoints, while state lists determine the dimensions of pairwise tables. Conditional extraction tasks receive reference components during SFT. The reverse task teaches the same model to describe a graph, providing the graph-to-text mapping used during reward refinement. Reference graph-to-text inputs also include available root priors. Each graph contributes $|E| + 4$ examples without task-specific reweighting, so larger graphs contribute more probability examples.

The shared model is trained with token cross-entropy on chat-formatted examples. The configuration does not construct an explicit assistant-only loss mask. We use rank-16 adapters, scaling $\alpha = 32$, and dropout 0.05 over attention and feed-forward projections, with frozen 4-bit base weights. Appendix A gives optimization settings, and Appendix B gives the instruction templates.

4.2 Graph-level reward refinement

For each training description x, the SFT-initialized policy samples $K = 4$ candidates $\widehat{G}^{(1)}, \ldots, \widehat{G}^{(K)}$. Each completion contains nodes, states, edges, and pairwise tables in one JSON object. This differs from the separate extraction completions used during SFT and evaluation. The reward assesses each candidate through its agreement with reference components, the description recoverable from it, and its represented size. These signals connect component accuracy, textual reconstruction, and complexity within one policy update.

Reference fidelity. A semantic judge aligns predicted variables and states to the reference. Component F1 scores measure node, state, and mapped directed-edge recovery. Let D_{CPD} be the mean reference-to-prediction KL over eligible columns of correctly recovered pairwise tables. The fidelity reward is

$$R_{\mathrm{fid}} = \frac{1}{4}\left(F_1^{\mathrm{node}} + F_1^{\mathrm{state}} + F_1^{\mathrm{edge}} + e^{-D_{\mathrm{CPD}}}\right). \tag{4}$$

The training state score uses graph-wide state counts, with matches restricted to aligned variables. The staged evaluator instead averages state scores over aligned variables. Unaligned endpoints are excluded from the mapped-edge score in both cases. Appendix A.1 describes matching and probability handling.

Semantic reconstruction. The current shared model verbalizes $\widehat{G}$ to produce $\widehat{x}$. A judge compares x and $\widehat{x}$, assigning $s \in \{1, 2, 3, 4, 5\}$ for agreement in variables, states, relationships, and their expressed strength. We rescale this score as

$$R_{\mathrm{cycle}} = (s - 1)/4. \tag{5}$$

The reconstruction prompt receives the candidate graph without the source description, so agreement supplies feedback on content recoverable through the graph. The reconstruction contributes a scalar reward, with no differentiable loss through verbalization. It provides a text-based assessment alongside direct reference matching, although verbalization and judging can introduce errors. Neither term by itself establishes numerical validity or a coherent joint probability model.

4.3 Reference-relative complexity control

Structural decisions also determine how much probability information a candidate represents: added variables, states, or edges can enlarge its collection of pairwise tables. We therefore count nodes, edges, states, and represented probability parameters across the entire candidate:

$$N(G) = |V|, \quad L(G) = |E|, \quad S(G) = \sum_{v \in V} |\mathcal{S}_v|, \quad D(G) = \sum_{Q \in \mathcal{Q}} (r_Q - 1)c_Q. \tag{6}$$

Nodes and edges are counted uniquely, and states are deduplicated within each node. A matrix with r_Q rows and c_Q normalized columns has $(r_Q - 1)c_Q$ free entries. A 3×2 table, for example, has four. This counts the represented pairwise matrices without assuming a joint BN parameterization. An edge incident to an unmatched variable is excluded from the mapped-edge comparison but still contributes to the candidate's edge count. Node F1 already penalizes unmatched variables, so these counts complement the fidelity signal.

For $M \in \{N, L, S, D\}$, define the positive relative surplus

$$\delta_M(\widehat{G}, G) = \max\left(0, \frac{M(\widehat{G}) - M(G)}{\max(M(G), 1)}\right), \qquad P_{\text{excess}} = \min\left(1, \sum_M w_M \delta_M\right), \tag{7}$$

with $(w_N, w_L, w_S, w_D) = (0.15, 0.35, 0.15, 0.35)$. For positive reference counts, normalization measures surplus in proportion to the corresponding reference count. Counts below the reference incur no excess penalty, while missing components can still reduce fidelity. The term measures cardinality surplus, rather than whether each component is supported by the description.

The reported reward configuration is

$$R = \text{clip}_{[0,1]}\left(0.5R_{\text{fid}} + 0.5R_{\text{cycle}} - \lambda P_{\text{excess}}\right). \tag{8}$$

We compare $\lambda = 0$ and $\lambda = 0.05$, retaining both fidelity and reconstruction terms. The penalty subtracts at most 0.05 before clipping. A completion with no recoverable JSON object receives zero reward. Other schema errors are handled within the scoring routines, without a separate DAG-validity gate.

Policy update. GRPO compares candidates for the same description by centering and scaling their rewards within the group. Candidates scoring above the group mean receive positive advantages, and those below it receive negative advantages. A clipped likelihood-ratio update uses these advantages to increase the relative likelihood of better-scoring completions (Shao et al., 2024). The reported sampling settings are temperature 0.9 and top-$p = 0.95$. Reward refinement changes the shared adapter through extraction completions. It does not introduce a separate graph-to-text policy loss. Table 2 contrasts the output decompositions used in training and evaluation.

5 Evaluation Protocol

Reference-assisted component evaluation. The evaluator queries nodes, states, edges, and pairwise tables in sequence. Node generation receives the description, with its output-token limit set from the reference node count. Semantic alignment uses Llama-3.3-70B-Instruct to match predicted node and state names to the reference. Reference matches also determine downstream inputs. State queries retain matched node names and their duplicate semantic matches. Edge queries use canonical matched nodes with predicted states, and probability queries are issued only for correctly recovered edges. The scores therefore measure component recovery under reference-assisted evaluation.

Node F1 discounts predictions classified as duplicate matches to an already recovered reference variable. State F1 is averaged over aligned variables. Edge F1 compares directed edges after endpoint mapping and excludes predictions with unaligned endpoints. A higher Node F1 can therefore reflect fewer unmatched predictions without recovering additional reference nodes.

Pairwise KL compares reference and predicted probability columns positionally on eligible recovered edges. Predictions are normalized and smoothed by 10^{-6}. Dimension handling can truncate, pad, or skip entries. Semantic state matches do not reorder the matrices. Bidirectional KL averages forward and reverse evaluations, applying preprocessing in each direction. The evaluator averages eligible columns within edges, then edges within examples, and reports the mean of available example scores. Missing metric values are omitted. Different models can thus contribute different sets of nodes, edges, and probability columns to an aggregate. Appendix A.2 specifies the scoring rules.

6 Results

Table 3 compares supervised initialization with two DUAL-BN refinement configurations, using GRPO with and without the excess-count penalty. We evaluate on 73 descriptions from five held-out source backbones. All three configurations use DeepSeek-R1-Distill-Qwen-14B and the same staged evaluation protocol. SFT uses extraction and graph-to-text examples. Both GRPO configurations retain fidelity and reconstruction rewards. The unadapted base model did not produce scorable outputs under this protocol and is therefore omitted from the table.

Stage	Conditioning information	Generated output	Supervision or scoring
SFT	Text and task-specific reference components, or a reference graph	One component type, one pairwise table, or a description	Token cross-entropy
GRPO	Description only for extraction	All graph components in one completion	Reference fidelity, reconstructed-text agreement, and excess counts
Evaluation	Text and reference-filtered predictions from earlier passes	Nodes, then states, edges, and tables	Conditional component metrics

Table 2: The model produces different output decompositions during SFT, GRPO, and staged evaluation. Reference components enter SFT prompts directly. During evaluation, reference matching determines which predicted components enter subsequent prompts.

Training	Node F1 ↑	Conditional State F1 ↑	Mapped Edge F1 ↑	Matched-edge pairwise KL ↓	Bidirectional KL ↓
SFT	0.7745	0.8738	0.6762	0.8707	0.6160
DUAL-BN ($\lambda = 0$)	0.8082	**0.8837**	0.6855	0.7189	**0.5302**
DUAL-BN ($\lambda = 0.05$)	**0.9106**	0.8486	**0.8415**	**0.6594**	0.5357

Table 3: Text-to-graph extraction on held-out backbones under reference-assisted evaluation. The two DUAL-BN variants refine the SFT policy with fidelity and reconstruction rewards. The full configuration uses $\lambda = 0.05$ for excess-count control. State F1 uses aligned variables, mapped Edge F1 excludes unaligned endpoints, and KL uses eligible probability columns on recovered edges. Node scoring discounts duplicate semantic matches. Means omit missing values independently, so the components contributing to each score can differ across models. Bold identifies the best mean in each column.

From supervised initialization to graph-level refinement. SFT produces predictions with Node F1 of 0.7745 and mapped Edge F1 of 0.6762. GRPO without the penalty improves all five reported means: Node F1 reaches 0.8082, mapped Edge F1 reaches 0.6855, and forward pairwise KL decreases from 0.8707 to 0.7189. Thus, reward refinement improves the assessed components beyond the supervised initialization, although its edge improvement over SFT is small. With the excess-count penalty, DUAL-BN reaches Node F1 of 0.9106 and mapped Edge F1 of 0.8415, gains of 0.1361 and 0.1653 over SFT, with forward KL of 0.6594.

Contribution of excess-count control. The regularized configuration has the largest node and edge gains. Relative to GRPO without the penalty, Node F1 is higher by 0.1024 and mapped Edge F1 by 0.1560. These results support accounting for excess represented size alongside component matching and reconstruction. The term measures a different property from fidelity. An incorrect edge can replace a reference edge without changing the count. Such a substitution incurs no excess penalty but can reduce fidelity. Conversely, an unmatched component can increase represented size even if it is excluded from a conditional comparison. The penalty supplements the matching reward, without guaranteeing smaller or semantically supported outputs in every case.

State and probability scores. Conditional State F1 is 0.8837 without the penalty and 0.8486 in the regularized configuration. Because state scoring depends on node alignment, the models need not be scored on the same variables. The aggregate cannot distinguish a change in the evaluated variables from a change in state-prediction quality on shared variables.

Forward pairwise KL decreases from 0.7189 to 0.6594, while bidirectional KL increases slightly from 0.5302 to 0.5357. Both metrics cover only eligible recovered edges and compare probability columns positionally. They measure numerical agreement on those comparisons, not the fraction of reference probabilities recovered. Published PRISM-BN generator scores use full conditional tables and are not matched baselines for this pairwise representation (Bhattacharya et al., 2026).

Changes in individual predictions. The selected cases in Appendix C distinguish several sources of higher Node F1. In Case 1, regularization accompanies additional reference recovery and fewer unmatched predictions. In Case 2, fewer unmatched variables raise precision, while the missing reference variable and edges remain absent. In Case 3, Node F1 increases because a predicted label is classified as a duplicate semantic match and removed from the precision denominator. The raw number of predictions and the recovered edges do not change. These cases were selected for improved Node F1 and illustrate possible behaviors without estimating their prevalence. They show why the aggregate F1 increase alone cannot establish that the model recovers more reference components or emits fewer raw predictions.

6.1 Composing partial representations

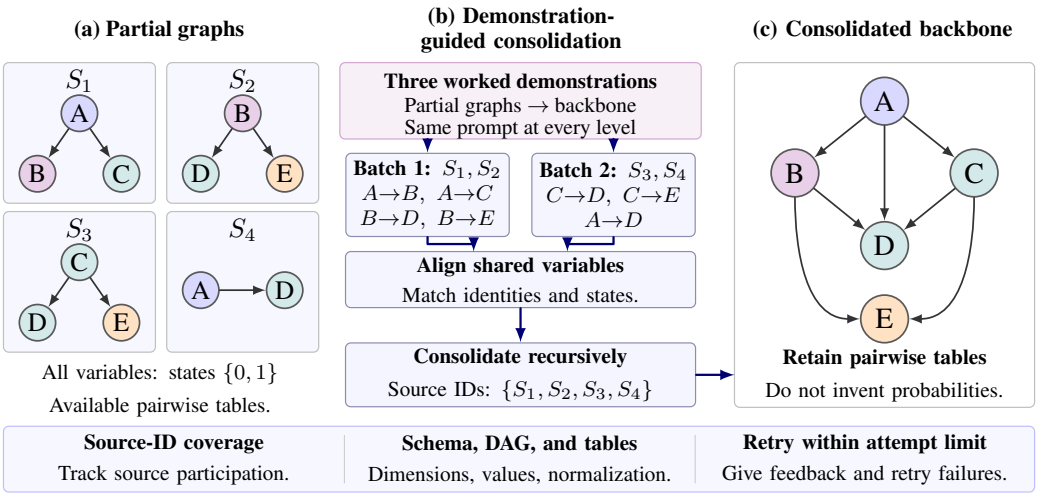


Figure 2: Composition from supplied partial graphs. Three demonstrations guide alignment and recursive consolidation. This illustrative example preserves the seven directed edges supplied across four partial graphs. The procedure tracks source identifiers and validates structures and included pairwise tables. More generally, the prompt permits additional components only with supporting justification. The automatic study uses supplied graphs.

Descriptions often cover only part of a larger system. We therefore examine, separately from extractor training, whether prompted models can consolidate supplied partial graphs into a backbone. With three worked demonstrations, GPT-5.6 attains Node, State, and Edge F1 of 0.907, 0.906, and 0.957 among completed reconstructions. These results show substantial structural overlap under the prompting procedure, alongside incomplete reference coverage. The available summaries do not establish a matched comparison across models. This automatic study uses supplied graphs and has no matched merging baseline.

Composition also exposes the distinction between recovering structure and preserving probability information. Two exactly recovered structures have full and zero probability-table coverage, respectively. A separate evaluation of descriptions generated from 39 reconstructed backbones gives a mean faithfulness score of 3.64 out of five, with unsupported claims remaining in many descriptions. Partial graphs can provide a basis for consolidation, but structural overlap alone does not establish faithful parameterization or verbalization. Backbone results appear in Appendix C.5, with consolidation and semantic evaluation protocols in Appendices D and E.

Human evaluation. The same three evaluators rated ten subgraph cases and ten backbone cases, with two cases per domain in each task. Each evaluator scored all four subgraph outputs per case. Mean overall ratings on a 1–5 scale increase from 3.41 for SFT to 3.79 after GRPO and 4.10 with regularization, compared with 4.44 for GPT-6 Astra. Regularized DUAL-BN receives 47% of the 30 preference choices, versus 43% for GPT-6 Astra and 10% for unregularized DUAL-BN. GPT-5.6 backbones composed from regularized DUAL-BN outputs receive an overall rating of $4.55/5$. These

judgments complement graph matching by assessing the meaning and appropriateness of recovered components. Appendices C.7 and F give the results and protocol.

7 Discussion and Limitations

DUAL-BN's gains over SFT support learning probabilistic graph components with structural and textual feedback. The full framework has the strongest node, mapped-edge, and forward-KL results, while unregularized GRPO has the strongest conditional State F1 and bidirectional KL. The experiments assess the combined framework and its complexity term without isolating the effects of bidirectional supervision, cycle feedback, or additional optimization.

The strongest node and edge improvements occur with reference-relative complexity control. Some fidelity terms exclude unaligned structure, whereas the penalty counts nodes, edges, states, and probability parameters across the candidate. This difference may help explain the gains, although the aggregate results do not isolate the mechanism. Counts complement fidelity but cannot distinguish graphs with different semantic content. The decline in conditional State F1 does not establish a decline on a fixed set of variables because scoring depends on node alignment.

Automatic extraction evaluation measures staged recovery using reference information to select downstream inputs. It therefore does not directly assess the joint graph completions produced during GRPO or establish extraction quality without reference assistance. State, edge, and probability scores can cover different recovered components across models, and probability scores do not measure coverage of all reference tables. Training rewards and structural evaluation also rely on model-based semantic alignment and inherit errors in those judgments.

Human evaluators rate the full framework above SFT and unregularized GRPO on all five criteria. GPT-6 Astra receives higher probability and overall ratings, although DUAL-BN receives a slightly larger share of preference choices. These comparisons cover ten descriptions and three evaluators, limiting conclusions about broader model differences.

Our pairwise target omits root priors and joint parent effects, so it does not specify a complete general BN. The partition tests transfer to five held-out PRISM-BN backbones under the corpus's text-generation procedure. Transfer to naturally occurring descriptions and other graph families remain untested.

Exact structure recovery can coexist with missing probability tables and unsupported claims in generated descriptions. The automatic composition study uses supplied graphs and has no matched consolidation baseline. Human evaluation of ten backbones constructed with DUAL-BN and GPT-5.6 assesses the combined pipeline, whose structural accuracy, probability coverage, and verbalization require broader evaluation.

8 Conclusion

We introduced DUAL-BN, a two-stage framework for learning probabilistic graph components from natural language. Bidirectional supervision connects extraction with graph-to-text reconstruction, and GRPO refines joint predictions through reference fidelity, semantic reconstruction, and reference-relative complexity control. On held-out PRISM-BN backbones, the full framework improves node and mapped-edge F1 over SFT by 0.1361 and 0.1653, respectively, and reduces forward pairwise KL from 0.8707 to 0.6594 under reference-assisted evaluation. These gains, together with the observed state and probability trade-offs, support learning from structured feedback alongside token-level supervision. Human evaluation adds an assessment of reconstruction quality, while the composition study examines how partial graphs can be assembled into larger models. Complete probability specifications and an integrated extraction-and-composition pipeline are next steps toward turning textual knowledge into probabilistic models.

AI Use Statement

Generative AI tools, including ChatGPT, assisted with manuscript drafting, editing, organization, and LaTeX formatting. The use of language models for extraction, verbalization, benchmark construction,

and automated evaluation is described separately in the paper and its cited data source. Writing assistance did not supply new empirical measurements. The authors are responsible for the reported results, citations, and interpretation.

Ethics Statement

The reference graphs and probabilities are constructed benchmark representations. Their dependencies should not be treated as established causal relationships, and their numerical values have not been validated for decisions about the systems they describe. The semantic evaluation includes examples in which generated text introduces causal or mechanistic interpretations unsupported by the reference. Applications of extracted models require independent assessment of these relationships and probabilities.

Reproducibility Statement

The paper specifies the representation, data partition, reported reward configuration, and evaluation conditions. Appendix A gives the optimization settings, Appendix A.2 details component scoring, and Appendices B and D document extraction and composition instructions. Appendix F gives the human-evaluation rubric, rating assignment, and aggregation, with model-specific results in Appendix C.7. PRISM-BN is described by Bhattacharya et al. (2026). Code and data have been made available for the reviewers and will be made open-source upon acceptance.

References

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 of the 31st International Conference on Computational Linguistics*, pp. 10685–10711, 2025. URL https://aclanthology.org/2025.coling-main.713/.

Amartya Bhattacharya, Nikhil Singh, Neeti Pokhriyal, and Soroush Vosoughi. PRISM-BN: A controlled corpus and benchmark for text-to-parameterized Bayesian network extraction. arXiv preprint arXiv:2609.21673, 2026. URL https://arxiv.org/abs/2609.21673.

Jiebin Cai, Yinghui Pan, Yifeng Zeng, and Han Liu. Bayesian network structure learning through large language models. In *Proceedings of the 25th International Conference on Autonomous Agents and Multiagent Systems*, pp. 3537–3539, 2026. doi: 10.65109/SGHT1267. URL https://doi.org/10.65109/SGHT1267.

Ruisheng Cao, Su Zhu, Chen Liu, Jieyu Li, and Kai Yu. Semantic parsing with dual learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 51–64, 2019. URL https://aclanthology.org/P19-1007/.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1feb87871436031bdc0f2beaa62a049b-Abstract.html.

Song Duong, Alberto Lumbreras, Mike Gartrell, and Patrick Gallinari. Learning from multiple sources for data-to-text and text-to-data. In *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pp. 3733–3753, 2023. URL https://proceedings.mlr.press/v206/duong23a.html.

Yu Feng, Ben Zhou, Weidong Lin, and Dan Roth. BIRD: A trustworthy Bayesian inference framework for large language models. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/19452e14fe6e0a8ac00410f1eebcbded-Abstract-Conference.html.

Jiaying Gu and Qing Zhou. Learning big Gaussian Bayesian networks: Partition, estimation and fusion. *Journal of Machine Learning Research*, 21(158):1–31, 2020. URL https://jmlr.org/papers/v21/19-318.html.

Qipeng Guo, Zhijing Jin, Xipeng Qiu, Weinan Zhang, David Wipf, and Zheng Zhang. CycleGT: Unsupervised graph-to-text and text-to-graph generation via cycle training. In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, pp. 77–88, 2020. URL https://aclanthology.org/2020.webnlg-1.8/.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. Dual learning for machine translation. In *Advances in Neural Information Processing Systems*, volume 29, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/5b69b9cb83065d403869739ae7f0995e-Abstract.html.

Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009. URL https://mitpress.mit.edu/9780262013192/probabilistic-graphical-models/.

Yassir Lairgi, Ludovic Moncla, Rémy Cazabet, Khalid Benabdeslem, and Pierre Cléau. iText2KG: Incremental knowledge graphs construction using large language models. arXiv preprint arXiv:2409.03284, 2024. URL https://arxiv.org/abs/2409.03284.

Wai Lam and Fahiem Bacchus. Learning Bayesian belief networks: An approach based on the MDL principle. *Computational Intelligence*, 10(3):269–293, 1994. doi: 10.1111/j.1467-8640.1994.tb00166.x. URL https://doi.org/10.1111/j.1467-8640.1994.tb00166.x.

Izhar Matzkevich and Bruce Abramson. The topological fusion of Bayes nets. In *Proceedings of the Eighth Conference on Uncertainty in Artificial Intelligence*, pp. 191–198, 1992. URL https://arxiv.org/abs/1303.5417.

Aliakbar Nafar, Kristen Brent Venable, and Parisa Kordjamshidi. Reasoning over uncertain text by generative large language models. arXiv preprint arXiv:2402.09614, 2024. URL https://arxiv.org/abs/2402.09614.

Aliakbar Nafar, Kristen Brent Venable, Zijun Cui, and Parisa Kordjamshidi. Extracting probabilistic knowledge from large language models for Bayesian network parameterization. arXiv preprint arXiv:2505.15918, 2025. URL https://arxiv.org/abs/2505.15918.

Timo Pierre Schrader, Lukas Lange, Simon Razniewski, and Annemarie Friedrich. QUITE: Quantifying uncertainty in natural language text in Bayesian reasoning scenarios. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 2634–2652, 2024. doi: 10.18653/v1/2024.emnlp-main.153. URL https://aclanthology.org/2024.emnlp-main.153/.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. arXiv preprint arXiv:2402.03300, 2024. URL https://arxiv.org/abs/2402.03300.

Zhuoer Wang, Marcus Collins, Nikhita Vedula, Simone Filice, Shervin Malmasi, and Oleg Rokhlenko. Faithful low-resource data-to-text generation through cycle training. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2847–2867, 2023. doi: 10.18653/v1/2023.acl-long.160. URL https://aclanthology.org/2023.acl-long.160/.

Lionel Wong, Gabriel Grand, Alexander K. Lew, Noah D. Goodman, Vikash K. Mansinghka, Jacob Andreas, and Joshua B. Tenenbaum. From word models to world models: Translating from natural language to the probabilistic language of thought. arXiv preprint arXiv:2306.12672, 2023. URL https://arxiv.org/abs/2306.12672.

Yinghuan Zhang, Yufei Zhang, Parisa Kordjamshidi, and Zijun Cui. Bayesian network structure discovery using large language models. arXiv preprint arXiv:2511.00574, 2026. URL https://arxiv.org/abs/2511.00574v2. Version 2.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E.

Gonzalez, and Ion Stoica. Judging LLM-as-a-Judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://papers.neurips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html.

A Training and Implementation Details

Supervised adaptation. The base model is DeepSeek-R1-Distill-Qwen-14B. We freeze its base weights using NF4 quantization with double quantization and bfloat16 computation. Rank-16 LoRA adapters use scaling $\alpha = 32$ and dropout 0.05. The adapted modules are the query, key, value, and output attention projections and the gate, up, and down feed-forward projections. The reported SFT configuration uses three epochs, learning rate 2×10^{-4}, a cosine schedule with 100 warmup steps, maximum sequence length 3,072, and per-device batch size one with one gradient-accumulation step. We use paged 8-bit AdamW, gradient checkpointing, and maximum gradient norm 1.0. Sequence packing is disabled. Validation loss is evaluated every 10,000 steps, when checkpoints are also saved; the checkpoint with the lowest evaluated validation loss is reloaded before saving the final adapter. Training uses seed 42.

Pairwise targets. The data loader reads each edge's stored `cpd_matrix` from the child node's parent record. It does not derive a pairwise table from a full multi-parent CPD. For SFT targets and reference graph serialization, nullable cells become zero and columns are normalized. A zero-sum column is replaced by a uniform distribution, and entries are rounded to four decimal places. The reward and evaluator read the stored matrices and apply their own probability preprocessing, described below.

Reward refinement. GRPO starts from the SFT adapter and uses the same quantized base model. The reported configuration allows up to 2,000 update steps, with learning rate 10^{-5}, a cosine schedule, 50 warmup steps, per-device batch size one, and four gradient-accumulation steps. Each description produces four sampled graph completions with temperature 0.9 and top-$p = 0.95$. Reverse generation uses the current model with the same sampling settings and a 600-token output limit. Llama-3.3-70B-Instruct supplies node and state alignment and the cycle score through the Hugging Face inference interface. Reverse generation is performed without gradients. The extraction completions receive the policy update.

A.1 Training reward computation

For m matches, n reference components, and k predicted components, precision is m/k and recall is m/n, with zero assigned when the corresponding denominator is zero. F1 is their harmonic mean, or zero when both are zero. The training scorer first extracts the JSON object from each completion. Unrecoverable JSON receives zero reward. Parsed node, edge, or table fields that are not lists are treated as empty lists. Other schema violations do not pass through a universal validity gate.

Nodes and states. Predicted node names are deduplicated before scoring. The node judge proposes semantic matches, and the scorer retains only matches involving existing labels with no repeated predicted or reference endpoint. States are deduplicated within nodes. The scorer first matches normalized exact state names within aligned node pairs, then asks the judge to match the remaining states and validates those matches for uniqueness. The state numerator counts accepted state matches within aligned variables. Its denominators count all states in the predicted and reference graphs, including states of unaligned variables. This differs from the conditional state score used in staged evaluation.

Edges. Predicted directed edges are mapped through the accepted node alignment. Edges with an unaligned endpoint are omitted from the mapped set, and repeated mapped edges are deduplicated. Edge F1 compares this set with all reference directed edges. Thus, unaligned edges do not directly enter the edge precision denominator, although they can affect the excess-count penalty.

Pairwise probabilities. Training compares tables only on correctly recovered edges. For each reference matrix, the number of available columns is the shortest row width. A prediction with fewer rows or available columns than the reference is skipped. Larger predictions are restricted to the reference dimensions. The comparison is positional; semantic state matches do not reorder rows or columns. Nullable cells become zero, and columns with numerical conversion errors are omitted.

Let p and q be one eligible reference and predicted column of length r. Define

$$\nu(z)_a = \begin{cases} z_a / \sum_j z_j, & \sum_j z_j > 0, \\ 1/r, & \text{otherwise}, \end{cases} \qquad \sigma_\epsilon(z)_a = \frac{\max(z_a, 0) + \epsilon}{\sum_j (\max(z_j, 0) + \epsilon)}, \quad \epsilon = 10^{-6}. \tag{9}$$

The implemented directional comparison is

$$d(p, q) = \sum_{a:\nu(p)_a > 0} \nu(p)_a \log \frac{\nu(p)_a}{\sigma_\epsilon(\nu(q))_a}. \tag{10}$$

The training quantity D_{CPD} is the mean over all eligible columns across recovered edges. If none is eligible, it is set to 5, giving a small positive e^{-5} contribution to the fidelity reward. These rules handle irregular predictions but do not constitute a complete probability-validity test.

Complexity counts and reconstruction. The excess penalty counts unique nonempty node labels, unique directed endpoint pairs, and unique state strings within each node. Its parameter count sums $\max(r - 1, 0)c$ over listed pairwise matrices, using the shortest row width for c. This is a count derived from represented dimensions, not a verification that columns are normalized or that the tables define a joint distribution. The reference count is computed from its parent-specific matrices. The weights, surplus normalization, cap, and reported final reward are given in Eqs. 6–8.

For the reconstruction term, the candidate is serialized as node–state lists followed by directed edges and their available pairwise matrices. The reverse model generates a description, and the judge compares it with the input description. A missing or unusable cycle judgment receives the lowest score, giving zero cycle reward. The reconstruction score is qualitative and does not establish exact preservation of probability entries.

A.2 Staged evaluation details

Evaluation uses greedy decoding and four successive extraction prompts. The node prompt receives the generated description, with an output budget of $\min(1000, 200 + 50|V|)$ tokens based on the reference node count. Positive semantic matches are grouped by reference node, with the first prediction serving as the canonical match. Additional predictions matched to the same reference are counted as duplicates. If d predictions receive this treatment and m reference nodes are matched, the node precision denominator is $\max(k - d, m)$, where k is the number of predicted node names. The output field `n_extra_nodes` stores d; it does not count unmatched variables.

The state prompt receives the canonical matched predictions and their duplicate semantic matches, excluding unmatched predicted nodes. The evaluator adds a `None` state when absent and reconciles returned node labels against the matched names. It computes state precision, recall, and F1 for each aligned variable and averages these scores within the example. Missing predicted states receive zero for that variable. If state judging fails, normalized exact matching provides a fallback. The staged scorer uses the positive judge matches without the training scorer's local one-to-one validation.

The edge prompt receives canonical matched node names and their predicted states. Edges are mapped through the node alignment, deduplicated, and compared with the reference edge set. Edges with unmatched endpoints are ignored. Probability prompts are issued only for correctly recovered edges and receive their predicted endpoint names and state lists. Consequently, reference matching affects both which components are scored and what the model is asked to predict.

Probability scoring and aggregation. The evaluator accepts two-dimensional numerical or nullable matrices. Nested cells and nonfinite or nonnumerical entries make a matrix unscorable. For a directional comparison, predicted rows are truncated or zero-padded to the reference child-state count. Missing or nullable cells become zero. Column iteration is bounded by the reference parent-state count and stops at the width of the prediction's first row. The evaluator applies Eq. 10 to available positional columns and averages them within each edge.

Write $\mathcal{L}(A, B; r, c)$ for this operation with predicted matrix A, reference matrix B, and reference dimensions (r, c). For each eligible edge, the reported bidirectional score is

$$\frac{1}{2}\left[\mathcal{L}(\widehat{Q}, Q; r, c) + \mathcal{L}(Q, \widehat{Q}; r, c)\right]. \tag{11}$$

The implementation stores this score as `cpd_kl_symmetric`. We call it bidirectional KL to avoid implying symmetry on a common support. An edge contributes to either reported KL aggregate only when both directional comparisons are available. Each direction performs its own normalization, smoothing, and dimension handling. When matrices have different dimensions, the two directions can use different available columns; this is not automatically a symmetric divergence on one common aligned support. No semantic state permutation is applied.

Forward and bidirectional scores are averaged across eligible edges within an example, then across examples with available scores. Other reported metrics also omit missing values independently. These aggregation rules allow different configurations to be scored on different recovered components, even when the test descriptions are identical.

A.3 Phase 2 implementation templates

The following templates reproduce the extraction, reverse-generation, and cycle-judging instructions in the implementation. Placeholder values are inserted at inference time. The phrase "full Bayesian network" is retained as prompt text; the requested schema contains the pairwise representation defined in Section 3.

Joint extraction: system instruction.

```
You are an expert at extracting Bayesian Networks from natural language.
Given a text, extract the full Bayesian network as JSON: nodes (with
states), directed
edges, and for each edge a CPD matrix (rows = child states, columns =
parent states,
each column sums to 1.0). Return ONLY valid JSON, no explanation.
```

Joint extraction: user template.

```
Extract the full Bayesian Network from this text.

Text:
"""{text}"""

Return ONLY this JSON structure:
{
  "nodes": [{"node": "name", "states": ["s1", "s2", "none"]}],
  "edges": [{"parent": "name", "child": "name"}],
  "cpds":  [{"parent": "name", "child": "name",
             "matrix": [[...]]}]
}

Rules:
- Every node has a "none" state
- Edges use only node names from the nodes list
- Each CPD matrix: rows = child states, columns = parent states, each
column sums to 1.0
```

Reverse generation: system instruction.

```
You are given a parameterized Bayesian Network. Generate a
natural-language
description (~350 words) mentioning every variable, every non-"none"
state, every causal
relationship, and conveying the STRENGTH of each relationship based on
the probability
magnitudes (strong vs weak influence).
```

Reverse generation: user template.

```
Given the following Bayesian Network, write a faithful natural-language
description.

{pgm_text}

Mention every variable, its states, every causal edge, and how strong
each influence is.
```

Cycle judge: system instruction.

```
You evaluate whether a regenerated description faithfully encodes the
same Bayesian network as an original description. Score 1 to 5. Return
ONLY valid JSON.
```

Cycle judge: user template.

```
Original description:
"""{x}"""

Regenerated description:
"""{x_hat}"""

Score how faithfully the regenerated description captures the SAME
Bayesian network as the
original, considering:
- Same variables (nodes) mentioned?
- Same states/values for each variable?
- Same causal relationships (which variable influences which)?
- Is the STRENGTH of each relationship conveyed consistently (strong vs
weak, matching the
  probability magnitudes)?

Scoring:
5 = fully faithful (variables, states, edges, and strengths all match)
4 = mostly faithful (minor omissions)
3 = partially faithful (some missing or strengths off)
2 = largely unfaithful (major structure missing)
1 = unfaithful (different/unrecognizable network)

Return ONLY: {"score": <1-5>, "reason": "<one short sentence>"}
```

B PROMPT TEMPLATES

The following templates specify the four staged extraction queries and the graph-to-text supervision task. Each query contains a system prompt and a user prompt. The complete-graph GRPO rollout is a separate query format described in Section 4.2. In the templates, `{TEXT}` is the description, `{NODES}` and state lists are JSON lists, `{NODE_STATES}` has one line `- node: [states]` per variable, and `{PGM}` is the serialized graph. Pass 4 is issued once per directed edge. Lines are wrapped for display.

These instructions include two conventions relevant to interpretation. The state prompt requires a `None` state for every variable. The edge prompt uses causal language. We preserve that wording below, but the evaluation measures agreement with benchmark dependencies and does not independently establish causation.

Pass 1 (variables). System:

```
You are an expert at identifying nodes (variables) in Bayesian Networks
↪  from natural
language.
Extract only the node names. Return ONLY valid JSON, no explanation.
```

User:

```
Extract all Bayesian Network nodes (variables) from this text.

Text:
"""{TEXT}"""

Return ONLY:
{"nodes": ["node1", "node2", "node3"]}

Rules:
- List every variable/node mentioned as a random variable
- Use exact names from the text
- Do not include states, only node names
```

Pass 2 (states). System:

```
You are an expert at identifying states (possible values) of Bayesian
↪  Network nodes from
natural language.
For each node given, extract all possible states. Always include a 'None'
↪  state for
every node.
Return ONLY valid JSON, no explanation.
```

User:

```
For each Bayesian Network node listed, extract all possible states from
↪  this text.

Text:
"""{TEXT}"""

Nodes: {NODES}

Return ONLY:
{
  "node_states": [
    {"node": "node name", "states": ["state1", "state2", "None"]}
  ]
}

Rules:
- Include every node from the list above
- List all states mentioned in the text for each node
- Always include "None" as a state for every node
```

Pass 3 (edges). System:

```
You are an expert at identifying directed causal relationships between
↪  Bayesian Network
nodes.
Given nodes and their states, extract which nodes causally influence
↪  which others.
Return ONLY valid JSON, no explanation.
```

User:

```
Identify directed causal edges (A → B means A causes/influences B)
↪  between these nodes.

Text:
"""{TEXT}"""

Available nodes and their states:
```

```
{NODE_STATES}

Return ONLY:
{
  "edges": [
    {"parent": "cause node", "child": "effect node"}
  ]
}

Rules:
- Only use node names from the list above
- Only include edges supported by the text
- An edge parent→child means the parent causally influences the child
```

Pass 4 (CPDs). System:

```
You are an expert at extracting Conditional Probability Distributions
↪  (CPDs) from
natural language.
Given a parent-child node pair with their states, produce the CPD matrix.
Rows = child states, columns = parent states. Each column must sum to
↪  1.0.
Return ONLY valid JSON, no explanation.
```

User:

```
Extract the CPD matrix for this causal relationship from the text.

Text:
"""{TEXT}"""

Parent "{PARENT}" states (columns): {PARENT_STATES}
Child  "{CHILD}"  states (rows):    {CHILD_STATES}

Return ONLY:
{
  "parent": "{PARENT}",
  "child":  "{CHILD}",
  "matrix": <{N_ROWS}x{N_COLS} list of lists, each column sums to 1.0>
}

Rules:
- Matrix is {N_ROWS} rows x {N_COLS} columns
- Row order matches child states exactly: {CHILD_STATES}
- Column order matches parent states exactly: {PARENT_STATES}
- Every column must sum to 1.0
```

Graph-to-text (training only). System:

```
You are given a parameterized Bayesian Network. Generate a
↪  natural-language description
(approximately 350 words) that mentions every variable, every non-"none"
↪  state, and
every causal relationship.
Use language strength that reflects the conditional probability
↪  magnitudes.
```

User:

```
Given the following parameterized Bayesian Network, generate a
↪  natural-language
description (~350 words) that mentions every variable, every non-"none"
↪  state, and every
causal relationship.

{PGM}
```

C SELECTED EXTRACTION EXAMPLES

We compare supervised fine-tuning (SFT), GRPO without the excess-count penalty, and GRPO with the penalty. The four cases were selected for higher Node F1 under regularization. They distinguish recovery of missing reference components, fewer unmatched predictions, and changes due to duplicate-aware scoring. They do not estimate how often these behaviors occur. Cases 1, 2, and 4 come from the same source backbone.

The graph panels show the aligned portions of the predictions, whereas the tables report complete predicted-node counts and reference matches. Unmatched predictions and duplicate semantic matches are absent from these panels; their omission from the drawings does not imply their absence from the raw output. Edge scores are conditional on node alignment and exclude edges with unaligned endpoints. The examples concern structure; they do not establish the accuracy of state spaces or probability tables. Relationships named below are entries in the reference graphs, rather than independently validated causal claims.

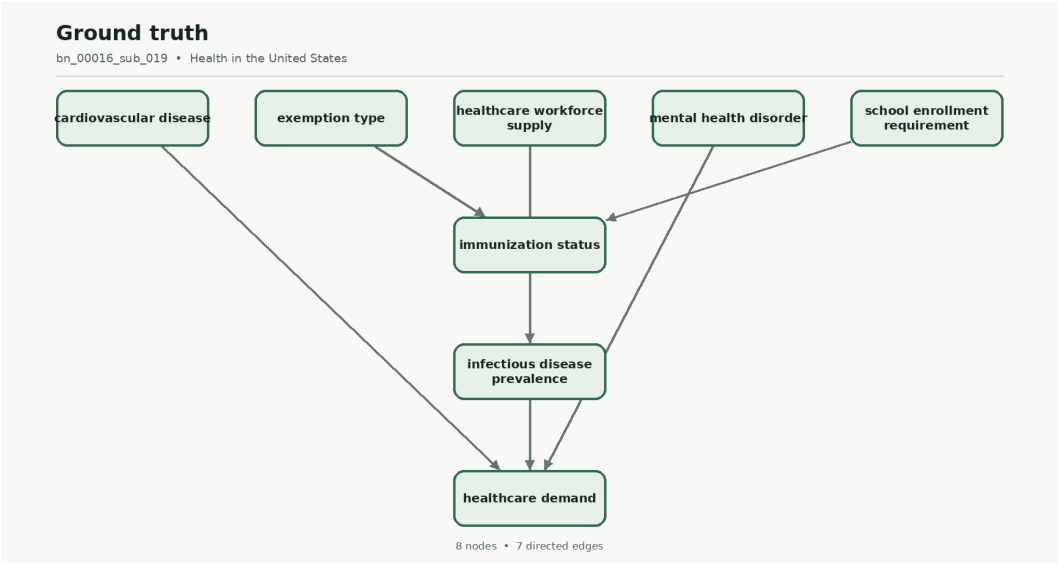
C.1 CASE 1: RECOVERY OF A MISSING VARIABLE AND EDGE

Case `bn_00016_sub_019` comes from the *Health in the United States* backbone. The reference contains eight variables and seven directed edges. Both SFT and unregularized GRPO recover seven reference variables and six reference edges. They miss *cardiovascular disease* and its outgoing edge to *healthcare demand*, and each predicts three unmatched variables. Regularized GRPO recovers the missing variable and edge while reducing the total predicted-node count from ten to eight (Table 4). The gain combines higher reference coverage with fewer unmatched predictions.

Metric	SFT	GRPO	GRPO + penalty
Predicted nodes	10	10	8
Matched reference nodes	7/8	7/8	8/8
Unmatched predicted nodes	3	3	0
Node precision	0.7000	0.7000	1.0000
Node recall	0.8750	0.8750	1.0000
Node F1	0.7778	0.7778	1.0000
Recovered reference edges	6/7	6/7	7/7
Conditional Edge F1	0.9231	0.9231	1.0000

Table 4: Case 1: more complete structural recovery with fewer unmatched variables. Conditional edge precision is 1.0000 for all variants. Unmatched-node counts are predicted nodes minus reference matches.

Reference



SFT: aligned prediction

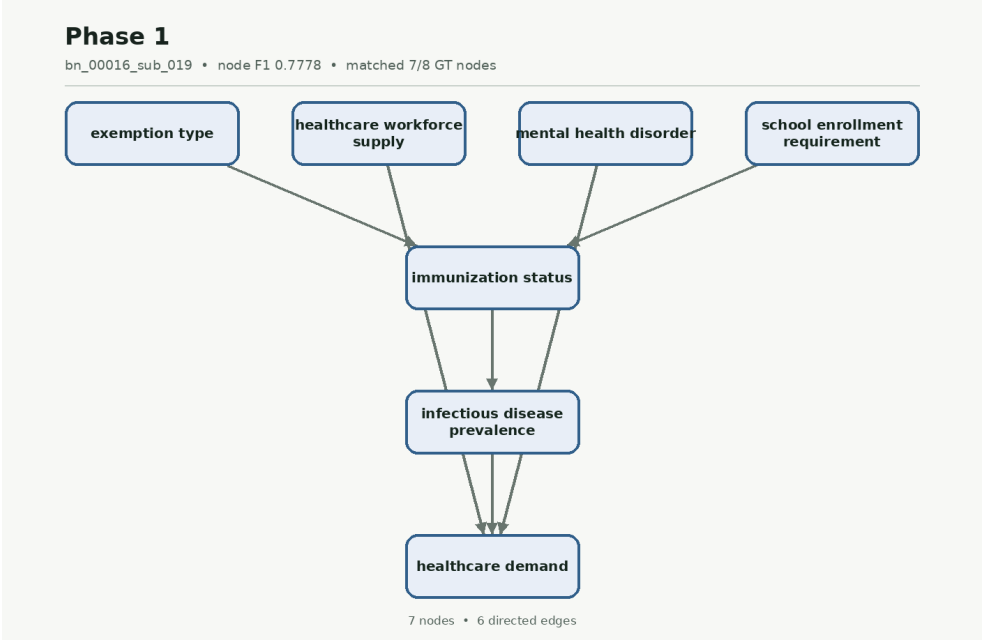
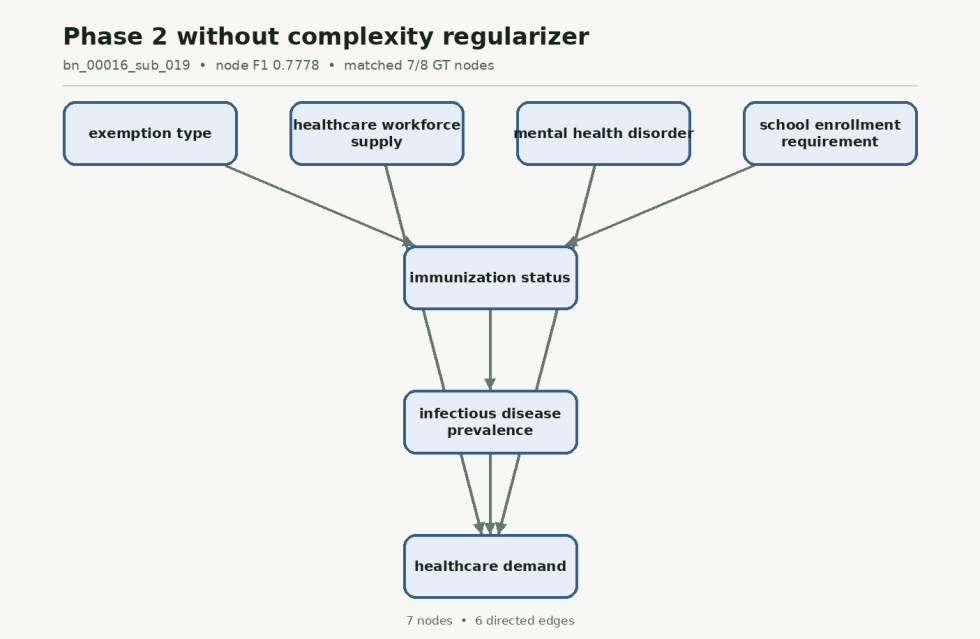


Figure 3: Case 1: reference (top) and SFT (bottom). The reference has eight nodes and seven edges. The SFT panel shows seven matched nodes and six recovered edges, omitting *cardiovascular disease* and its edge to *healthcare demand*. Three unmatched predictions are absent from the drawing; the complete output has ten nodes (Table 4).

GRPO: aligned prediction



GRPO with penalty: aligned prediction

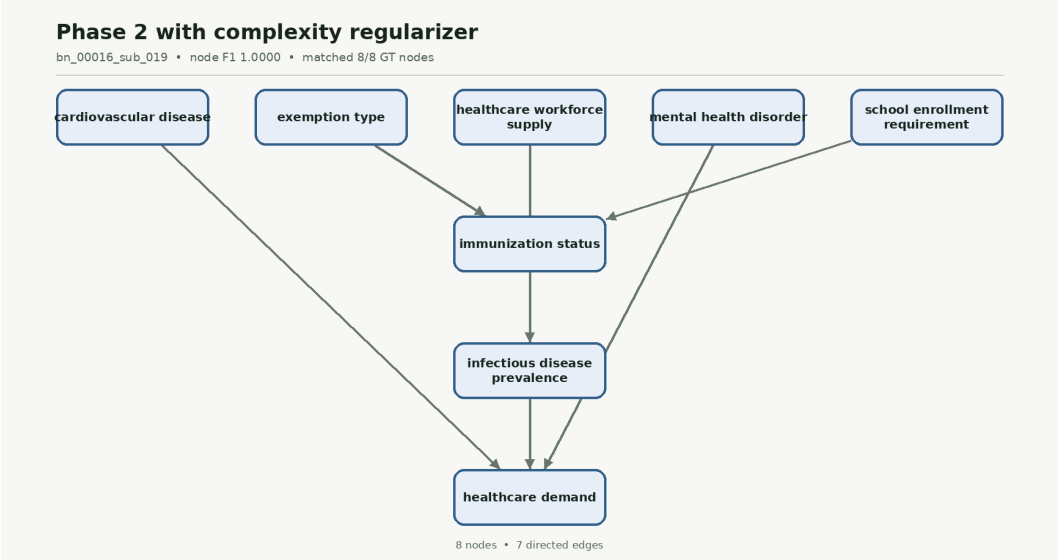


Figure 4: Case 1: GRPO without (top) and with (bottom) the penalty. The unregularized panel shows seven matched nodes and six recovered edges, with three unmatched predictions omitted. Regularized GRPO recovers all eight reference nodes and seven edges with eight predicted nodes in total. The panels display structure, not states or probability tables.

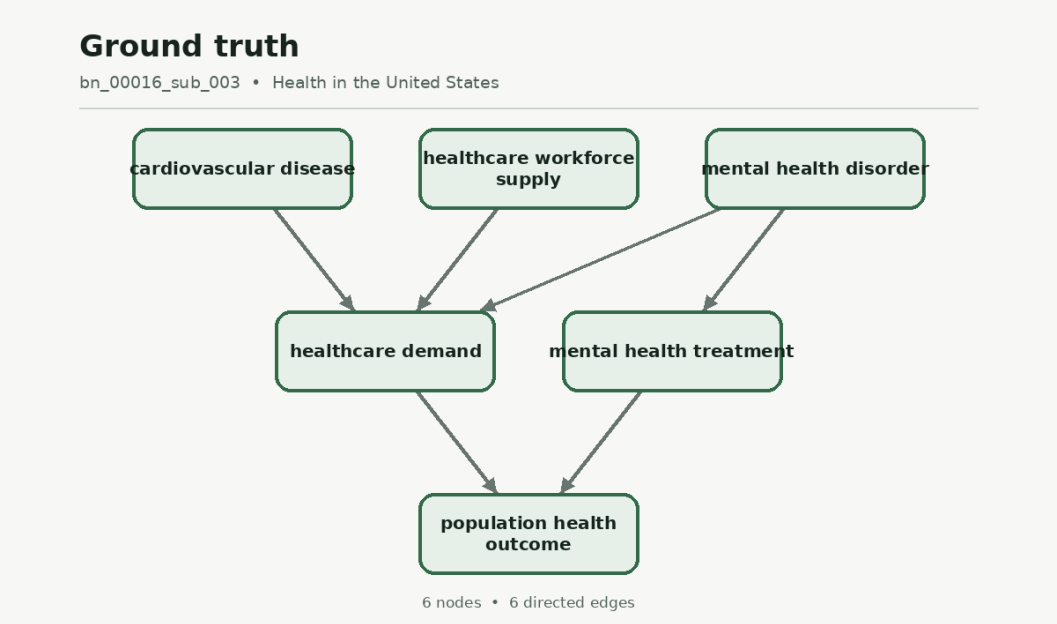
C.2 Case 2: Fewer unmatched variables with an unresolved omission

Case `bn_00016_sub_003` is another subgraph of the health backbone. The reference has six nodes and six edges. Every variant misses *population health outcome* and its two incoming edges from *healthcare demand* and *mental health treatment*. Regularized GRPO reduces the unmatched-node count from three to two, raising node precision from 0.6250 to 0.7143. Reference-node and edge recovery remain unchanged (Table 5). The improvement in Node F1 therefore reflects fewer unmatched variables, with the missing outcome and its relationships still unresolved.

Metric	SFT	GRPO	GRPO + penalty
Predicted nodes	8	8	7
Matched reference nodes	5/6	5/6	5/6
Unmatched predicted nodes	3	3	2
Node precision	0.6250	0.6250	0.7143
Node recall	0.8333	0.8333	0.8333
Node F1	0.7143	0.7143	0.7692
Recovered reference edges	4/6	4/6	4/6
Conditional Edge F1	0.8000	0.8000	0.8000

Table 5: Case 2: fewer unmatched predictions without additional reference recovery. Conditional edge precision is 1.0000 for all variants. There are no duplicate semantic matches.

Reference



SFT: aligned prediction

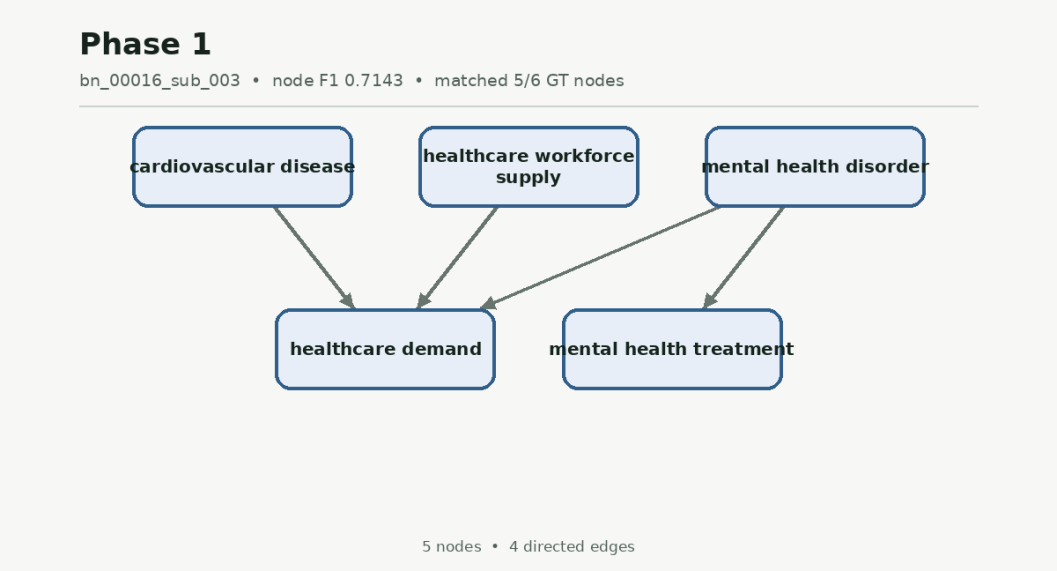
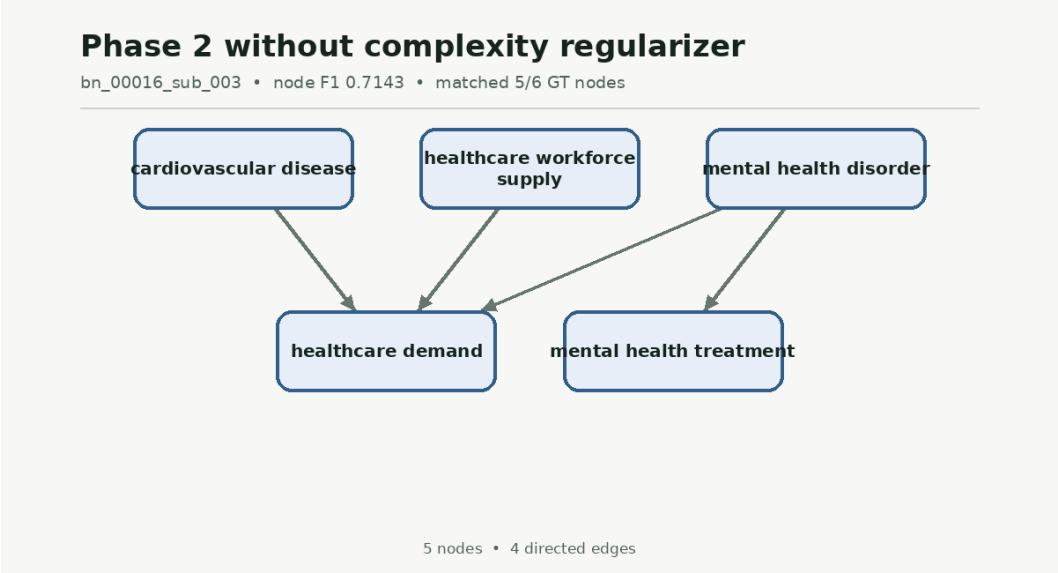


Figure 5: Case 2: reference (top) and SFT (bottom). The reference has six nodes and six edges. SFT recovers five nodes and four edges, missing *population health outcome* and its two incoming edges. Its three unmatched predictions are omitted from the drawing, which therefore shows five of eight predicted nodes.

GRPO: aligned prediction



GRPO with penalty: aligned prediction

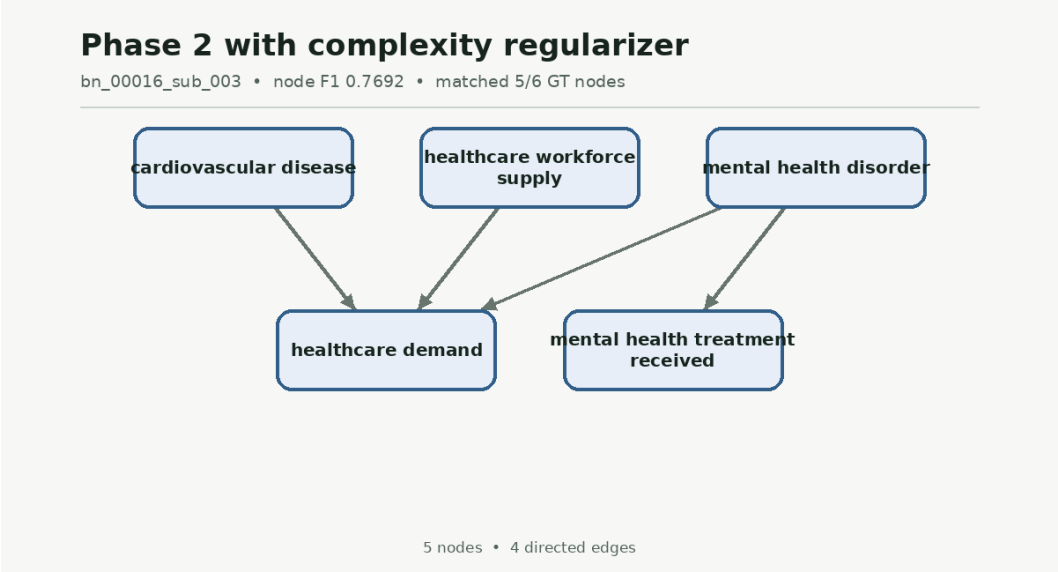


Figure 6: Case 2: GRPO without (top) and with (bottom) the penalty. Both panels show five matched nodes and four recovered edges. Their complete outputs contain eight and seven nodes, respectively; three and two unmatched predictions are not shown. Node F1 rises from 0.7143 to 0.7692, while the missing outcome and its two edges remain unresolved.

C.3 CASE 3: DUPLICATE-AWARE SCORING WITH UNCHANGED OUTPUT SIZE

Case `bn_00018_sub_021` comes from the *Climate change in Europe* backbone. All three variants match all six reference variables and recover all five reference edges. Their raw predictions each contain seven nodes. SFT and unregularized GRPO have one unmatched prediction. For regularized GRPO, the additional prediction is instead classified as a duplicate semantic match to an already matched reference variable.

The evaluator excludes such duplicate matches from the precision denominator. Let N_{pred}, N_{dup}, N_{match}, and N_{ref} denote raw predicted nodes, duplicate matches, canonical reference matches, and reference nodes. Then

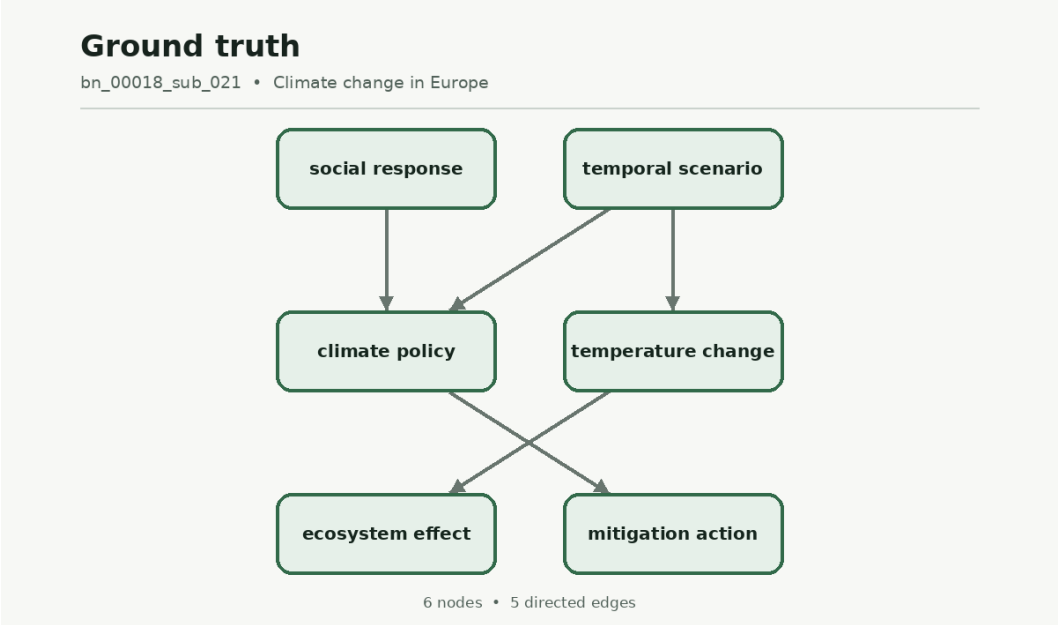
$$N_{\text{eff}} = \max(N_{\text{pred}} - N_{\text{dup}}, N_{\text{match}}), \qquad F_1^{\text{node}} = \frac{2N_{\text{match}}}{N_{\text{eff}} + N_{\text{ref}}}. \tag{12}$$

The reported Node F1 consequently increases from $12/(7+6) = 0.9231$ to $12/(6+6) = 1.0000$ (Table 6). The raw output size and reference coverage do not change. In the evaluation implementation, the field `n_extra_nodes` records duplicate semantic matches; it should not be interpreted as a count of unmatched predictions.

Metric	SFT	GRPO	GRPO + penalty
Raw predicted nodes	7	7	7
Canonical reference matches	6/6	6/6	6/6
Duplicate matches excluded	0	0	1
Effective predicted nodes	7	7	6
Unmatched predictions	1	1	0
Node precision	0.8571	0.8571	1.0000
Node recall	1.0000	1.0000	1.0000
Node F1	0.9231	0.9231	1.0000
Recovered reference edges	5/5	5/5	5/5
Conditional Edge F1	1.0000	1.0000	1.0000

Table 6: Case 3: duplicate-aware precision accounts for the Node F1 improvement. Unmatched predictions equal raw predictions minus canonical and duplicate matches. All three variants have identical reference coverage and raw node counts.

Reference



SFT: aligned prediction

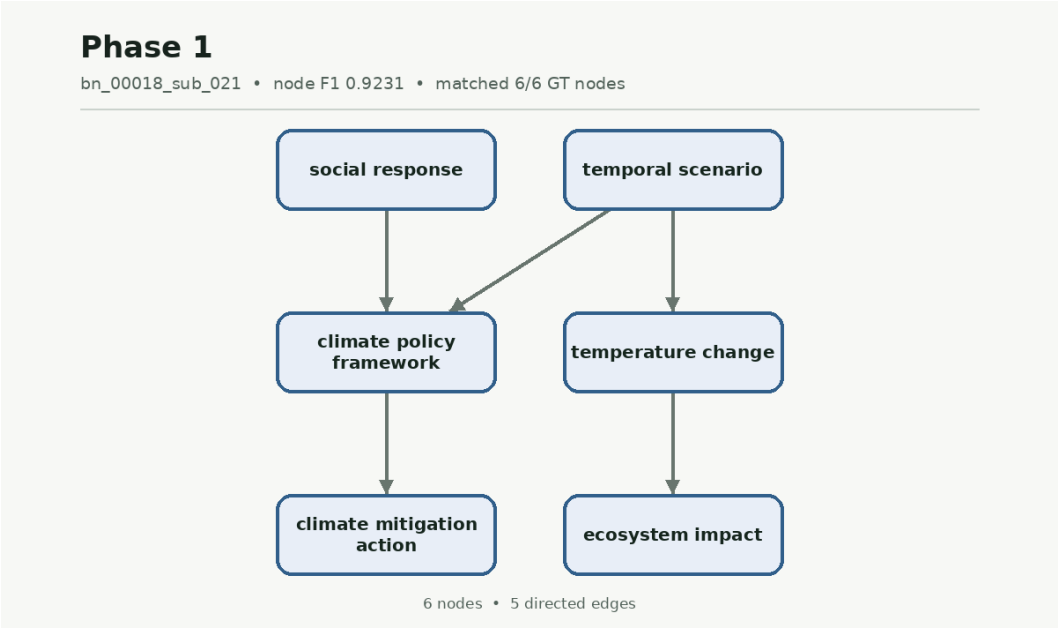
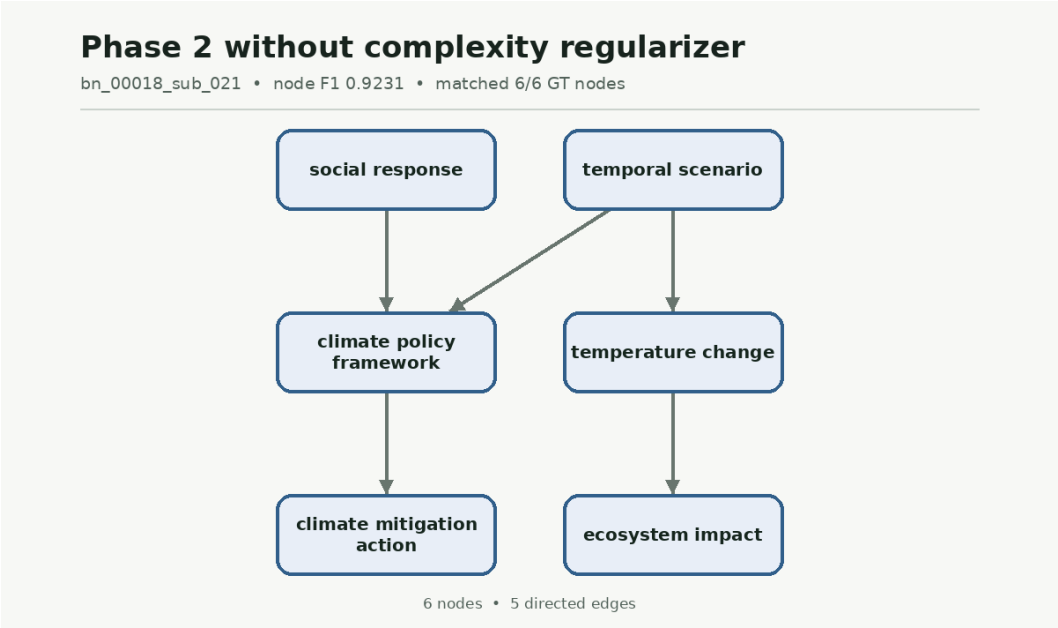


Figure 7: Case 3: reference (top) and SFT (bottom). SFT recovers all six reference nodes and five edges under semantic alignment. Its seventh, unmatched prediction is absent from the panel. The displayed labels need not exactly equal the reference labels.

GRPO: aligned prediction



GRPO with penalty: aligned prediction

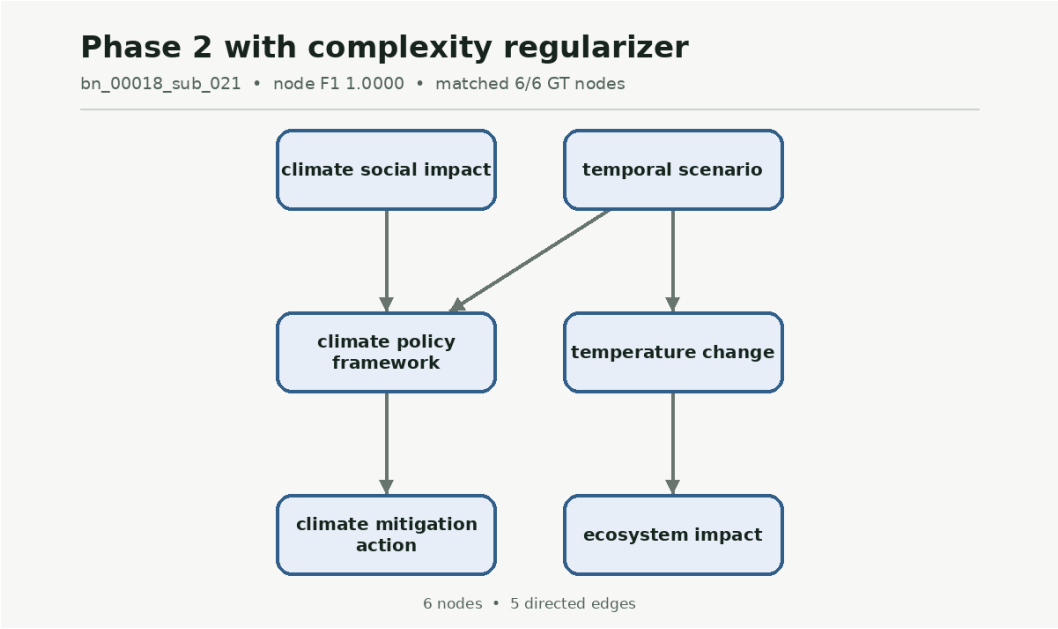


Figure 8: Case 3: GRPO without (top) and with (bottom) the penalty. Both raw outputs contain seven nodes and recover all six reference nodes and five edges. The omitted seventh prediction is unmatched for unregularized GRPO and a duplicate semantic match for regularized GRPO. The higher Node F1 reflects the duplicate-aware denominator, not fewer raw nodes or additional reference recovery.

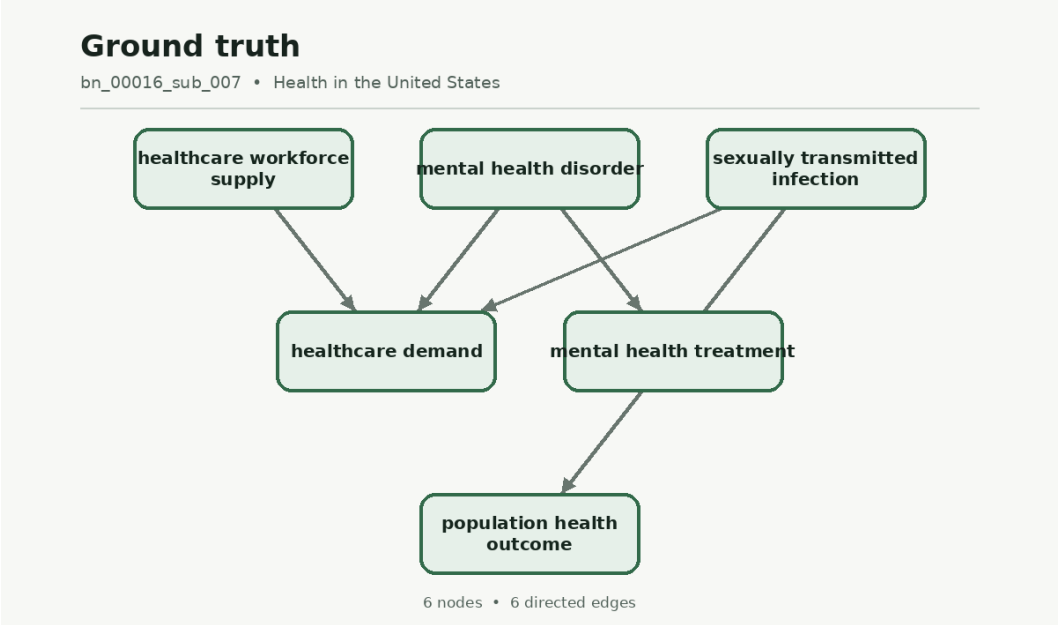
C.4 CASE 4: FEWER UNMATCHED VARIABLES WITH COMPLETE REFERENCE COVERAGE

Case `bn_00016_sub_007` is a third subgraph of the health backbone. All variants recover its six reference nodes and six reference edges. SFT and unregularized GRPO each predict seven nodes, including one unmatched variable. Regularized GRPO predicts six nodes, all matched to the reference, with no duplicate matches (Table 7). Here, the higher Node F1 accompanies a reduction in raw output size while reference coverage is preserved.

Metric	SFT	GRPO	GRPO + penalty
Raw predicted nodes	7	7	6
Canonical reference matches	6/6	6/6	6/6
Duplicate matches excluded	0	0	0
Unmatched predictions	1	1	0
Node precision	0.8571	0.8571	1.0000
Node recall	1.0000	1.0000	1.0000
Node F1	0.9231	0.9231	1.0000
Recovered reference edges	6/6	6/6	6/6
Conditional Edge F1	1.0000	1.0000	1.0000

Table 7: Case 4: removal of one unmatched prediction with complete reference-node and edge recovery. Raw and effective node counts coincide because there are no duplicate matches.

Reference



SFT: aligned prediction

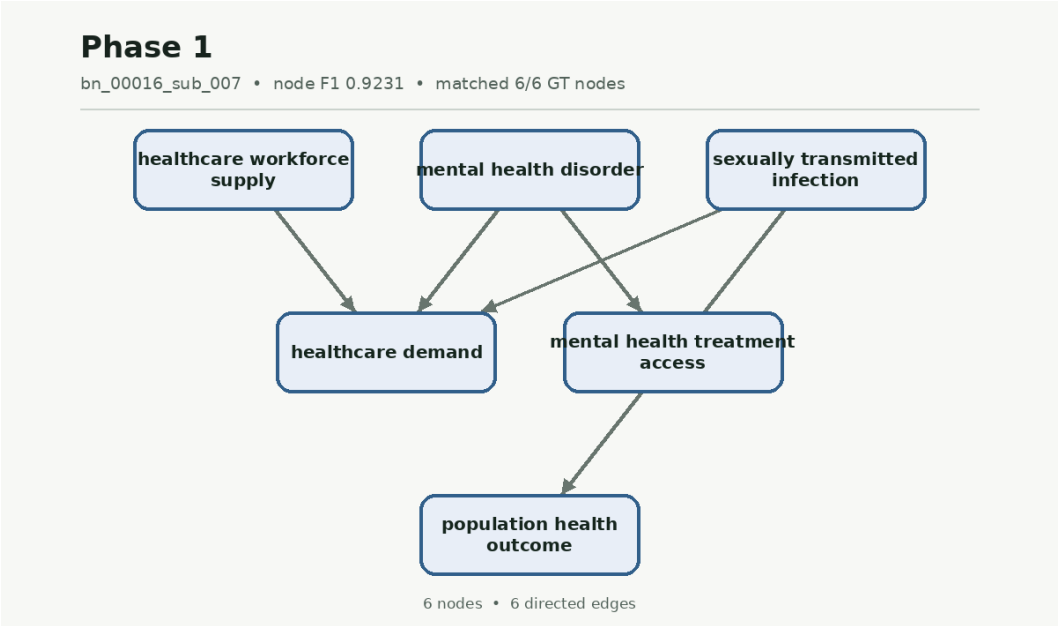
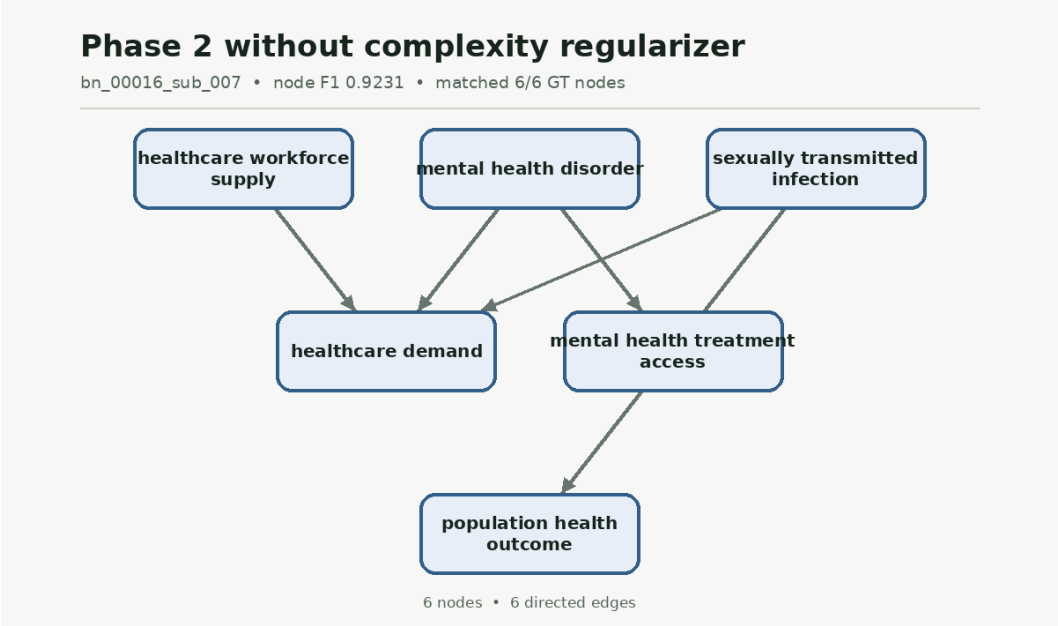


Figure 9: Case 4: reference (top) and SFT (bottom). Both panels show six nodes and six edges. SFT recovers all reference components under semantic alignment but also produces one unmatched node, absent from the drawing; its complete output contains seven nodes.

GRPO: aligned prediction



GRPO with penalty: aligned prediction

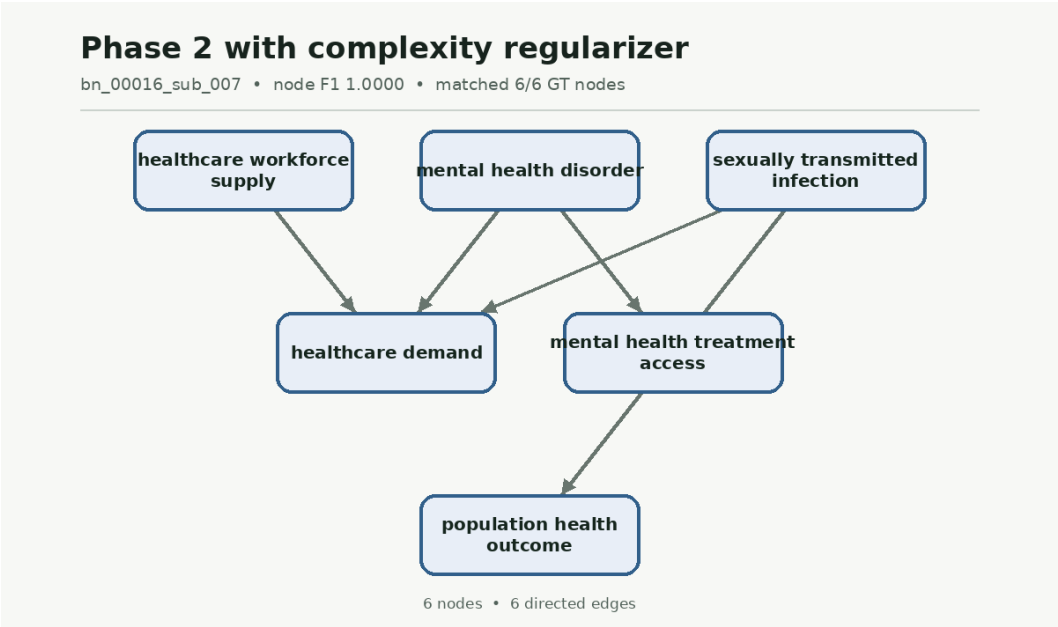


Figure 10: Case 4: GRPO without (top) and with (bottom) the penalty. Each panel shows all six reference nodes and six edges. The unregularized output has one additional unmatched node that is not displayed. Regularized GRPO has six predicted nodes and no duplicate matches; the node-count reduction is recorded in Table 7.

C.5 TASK 2: BACKBONE GENERATION

Task 2 evaluates whether language models can reconstruct a larger Bayesian network backbone by consolidating multiple partial subgraphs. All models are provided with the same set of three demonstrations and the same reconstruction instructions. We first evaluate backbone reconstruction using automatic structural metrics and execution validity. Because these metrics capture reference overlap but cannot fully assess the semantic quality of a reconstructed backbone, we complement this analysis with a human evaluation in Appendix C.7.

Table 8 summarizes structural reconstruction quality and execution failures across 47 evaluation cases (3 cases were provided examples in the prompt). GPT-5.6 and GPT-6 Astra each produce valid, evaluable outputs for 45 of the 47 cases, corresponding to a completion rate of 95.7%. The remaining two cases fail because the generated responses are incomplete or do not satisfy the required graph schema. Claude Opus 5 reconstruction scores come from five successful cases in an earlier run and are included only as a separate descriptive result. Reconstruction scores are reported only for successfully evaluated cases, while execution failures are reported separately.

Backbone reconstruction performance. Among completed cases, both GPT models recover the reference backbone structure with high F1. GPT-5.6 achieves Node, State, and Edge F1 scores of 0.9073, 0.9058, and 0.9569, respectively, while GPT-6 Astra achieves corresponding scores of approximately 0.904, 0.901, and 0.945. Both models recover the same proportions of reference nodes (0.8322), states (0.8293), and edges (0.9217), indicating that their differences in F1 arise primarily from precision rather than recall.

The strongest recovery is observed for graph relationships. GPT-5.6 reaches an edge precision of 0.9950, compared with 0.9685 for GPT-6 Astra, while both models recover the same proportion of reference edges. This suggests that, once the relevant variables are recovered, the consolidation procedure can preserve most supported relationships while introducing relatively few additional edges.

Backbone coverage. The automatic results also reveal an important distinction between structural precision and complete backbone recovery. Although Node and State F1 exceed 0.90 for both GPT models, node recall is 0.8322 and state recall is 0.8293. Thus, the reconstructed graphs tend to be precise but do not recover every variable and state in the reference backbone.

Edge recovery is stronger, with recall of 0.9217. This pattern suggests that relationships among recovered variables are preserved more consistently than the complete set of variables and states appearing in the reference network. The remaining challenge is therefore primarily one of coverage rather than large-scale introduction of unsupported structure.

Structured-output reliability. Task 2 also tests whether the models can consistently produce outputs that satisfy the required graph representation. GPT-5.6 and GPT-6 Astra fail on only 2 of the 47 evaluation cases, yielding valid outputs in 95.7% of cases. The available Claude reconstruction scores are a Node F1 of 0.8049, State F1 of 0.8013, and Edge F1 of 0.8916. However, these values are computed from five successful cases from an earlier evaluation. We report these results for completeness but do not use them for a directly matched reconstruction comparison with the two GPT models.

Beyond exact-match structural evaluation. Automatic F1 provides a reproducible measure of agreement with the reference backbone, but it does not capture every aspect of reconstruction quality. In particular, a component that does not exactly match a reference label may still represent semantically related information, while an automatically matched component may require domain-level judgment to determine whether it is appropriate in the reconstructed system. Similarly, structural overlap alone does not establish whether the resulting backbone forms a coherent and meaningful representation of the source material.

We therefore treat the automatic evaluation as one component of a broader assessment rather than as a complete measure of reconstruction quality. A separate human study assesses backbones composed by GPT-5.6 from regularized DUAL-BN outputs, as described in Appendix C.7. It examines reconstruction quality beyond reference overlap on a different set of inputs from the automatic study.

Model	Fail	Node F1 ↑	State F1 ↑	Edge F1 ↑
GPT-5.6	2	**0.907**	**0.906**	**0.957**
GPT-6 Astra	2	0.904	0.901	0.945
Claude Opus 5	—	0.805	0.801	0.892

Table 8: Automatic backbone reconstruction results. F1 scores are micro-averaged over successfully evaluated cases. The GPT models each complete 45 of the 47 target cases. The Claude F1 scores come from five successful cases in an earlier run and have no comparable failure count in this table. Higher F1 is better.

Appendix D.5 shows two exactly recovered backbones, one with full CPD coverage and one with none.

C.6 LLM-AS-A-JUDGE SEMANTIC FAITHFULNESS EVALUATION

Automatic graph metrics measure structural overlap with the reference backbone, but they do not directly assess whether the information represented by the generated graph remains semantically faithful when expressed in natural language. We therefore perform an independent graph-to-text faithfulness evaluation on the best-performing backbone model, GPT-5.6. For each successful backbone included in this assessment, GPT-5.6 converts its prediction into a standalone article-style description. GPT-6 Astra then evaluates this generated text against the untouched reference backbone. The judge receives only the reference graph and generated text; it does not observe the predicted backbone, structural metrics, target source article, or text-generation prompt.

GPT-6 Astra assigns a score from 1 to 5, where 5 indicates full agreement with no materially unsupported claims, 4 indicates mostly correct content with only minor inaccuracies, and 3 indicates a mixture of supported and unsupported claims. Importantly, the judge is instructed not to penalize omitted graph content, making this evaluation a measure of semantic faithfulness rather than backbone coverage. Coverage is measured separately by the structural metrics reported above.

The reported score distribution covers 39 descriptions. Across these cases, the mean judge score is 3.64 ± 0.74 out of 5, with a median of 3.0. All 39 descriptions receive a score of at least 3, and 19 cases (48.7%) receive a score of 4 or 5; six receive the maximum score of 5. The judge rationales show that the remaining errors are primarily semantic overstatements, such as expressing probabilistic dependencies as causal relationships or introducing relationships, mechanisms, or temporal claims not supported by the reference graph. Together with the structural evaluation, these results provide complementary evidence that the reconstructed backbones are not only close to the reference graph in node and edge space, but can also support largely faithful natural-language descriptions. Full generation and judging instructions, the scoring rubric, and representative rationales are given in Appendix E.

C.7 HUMAN EVALUATION

The same three evaluators completed both tasks. Each task contained ten cases, with two cases from each of the five domains. Every evaluator rated every case and every displayed model output, with no missing ratings. All criteria used a scale from 1 to 5, where 1 indicated poor or mostly incorrect reconstruction, 3 indicated mixed or partially correct reconstruction, and 5 indicated highly accurate reconstruction. Means average equally over examples and evaluators. Appendix F describes the interfaces and rubric.

Subgraph reconstruction. For each source passage, evaluators assessed one reconstruction from each of four models: SFT, DUAL-BN refined with GRPO without regularization, DUAL-BN refined with GRPO and regularization, and GPT-6 Astra. They rated nodes, states, edges, conditional probability tables, and overall quality, then selected a preferred reconstruction. Each model's mean for each criterion aggregates 30 ratings over the same ten source passages.

Table 9 shows higher node, state, edge, and overall ratings after GRPO refinement. Adding regularization raises all five means relative to unregularized GRPO and gives the highest overall rating among the three DUAL-BN configurations. Regularized DUAL-BN also receives the highest node, state, and edge ratings across the four models. GPT-6 Astra receives higher probability-table and

Model	Nodes	States	Edges	CPDs	Overall
SFT	3.38	3.58	3.49	3.36	3.41
DUAL-BN ($\lambda = 0$)	3.79	4.28	3.95	3.36	3.79
DUAL-BN ($\lambda = 0.05$)	**4.25**	**4.66**	**4.13**	3.89	4.10
GPT-6 Astra	4.01	4.60	4.03	**4.44**	**4.44**

Table 9: Mean human ratings of subgraph reconstructions on a 1–5 scale. Each cell averages 30 ratings from three evaluators on ten cases, with two cases per domain. Higher is better. Bold marks the highest mean in each column.

overall ratings. Probability tables remain the lowest-rated component for regularized DUAL-BN, consistent with the difficulty of recovering numerical information from text.

Across 30 preference choices, regularized DUAL-BN is selected in 47%, GPT-6 Astra in 43%, and unregularized DUAL-BN in 10% of choices, with percentages rounded to whole numbers. These are three judgments on each of ten cases. Regularized DUAL-BN has a slightly larger share of preferences, while GPT-6 Astra has the higher mean overall rating. The results support improved perceived reconstruction quality over SFT on the evaluated sample, without establishing a general advantage over GPT-6 Astra.

Backbone reconstruction. For this human study, GPT-5.6 composed regularized DUAL-BN outputs into backbones. GPT-5.6 was selected because it obtained the highest structural scores in the automatic composition study, with results close to GPT-6 Astra. Evaluators saw one reconstruction per case, together with the original source text and a natural-language description generated from the backbone. They rated node, state, edge, and overall quality using the same 1–5 scale. Each criterion has 30 ratings across ten cases and three evaluators.

Criterion	Mean rating
Nodes	4.14
States	4.77
Edges	4.02
Overall quality	4.55

Table 10: Human ratings of GPT-5.6 backbones composed from regularized DUAL-BN outputs. Each mean averages 30 ratings on a 1–5 scale. The ten cases include two from each domain.

State quality receives the highest mean rating at 4.77, followed by nodes at 4.14 and edges at 4.02, with overall quality rated 4.55. These judgments support the coherence of the reconstructed backbones in this sample. The human study assesses a pipeline using DUAL-BN predictions, whereas the automatic composition study uses supplied partial graphs. The assessments provide complementary views of reconstruction quality on small samples, with broader evaluation needed across other descriptions and systems.

D BACKBONE CONSOLIDATION PROTOCOL

The exploratory consolidation study operates on collections of partial reference graphs. It does not use the trained extractor's predictions. Model parameters remain fixed, and the target reference backbone is withheld during generation.

D.1 BACKBONE-RECONSTRUCTION PROMPT

All evaluated models receive the same reconstruction instructions, output contract, and three fixed few-shot demonstrations. The demonstrations pair collections of partial Bayesian-network graphs with their corresponding consolidated backbones.

The instructions below give the conceptual form of the reconstruction prompt. Graph serialization and the JSON schema are inserted programmatically, so this is not a complete serialized request.

> **Prompt: Backbone reconstruction**
>
> You are given multiple partial Bayesian-network graphs describing the same underlying system.
>
> Your task is to reconstruct a single coherent, maximal plausible Bayesian-network backbone from the supplied partial graphs.
>
> Follow the instructions below.
>
> 1. **Variables.** Treat nodes as variables. Merge node labels only when they identify the same underlying variable. Do not merge variables merely because they are semantically related.
> 2. **States.** Reconcile the state lists of matched variables. Preserve states supported by the supplied graphs and avoid introducing unsupported state distinctions.
> 3. **Directed relationships.** Preserve directed relationships supported by the supplied evidence. Edge direction must be retained whenever it is supported by the inputs.
> 4. **Additional supported components.** The output is not restricted to a literal union of the supplied graphs. You may recover an omitted node, state, or directed relationship when it is supported by the supplied evidence and the demonstrations.
> 5. **Probability information.** Preserve and reconcile probability information when the numerical values are justified by the supplied graphs. Do not invent numerical probabilities.
> 6. If a directed relationship is supported but its probability values cannot be justified, the relationship may be included without a probability table.
> 7. **Justification.** Include a concise justification for each inferred component.
> 8. **Graph validity.** The final graph must be a directed acyclic graph. Do not include self-loops or duplicate directed edges.
> 9. **Schema validity.** Node names and states must be unique within their respective scopes, and every edge endpoint must refer to a node included in the output.
> 10. **Output format.** Return only an object that conforms to the required JSON output schema.
>
> Construct the most complete backbone that can be justified from the provided evidence. Do not assume access to the withheld reference backbone.

Here, *maximal plausible backbone* is an operational description. It means that supported information beyond a literal union of the partial graphs may be recovered; it does not imply a unique or formally maximum graph under a graph-theoretic optimization criterion.

D.2 FEW-SHOT DEMONSTRATIONS

The same three demonstrations are supplied with every request. Each demonstration consists of a set of partial graphs and its corresponding consolidated backbone. The three demonstration backbones are excluded from the reconstruction targets.

Conceptually, each demonstration is serialized as

```
DEMONSTRATION INPUT

[
    {
        "source_id": <source identifier>,
        "graph": <partial Bayesian-network graph>
    },
    ...
]

DEMONSTRATION OUTPUT

<consolidated Bayesian-network backbone>
```

The demonstrations illustrate how overlapping variables, states, directed dependencies, and available probability information should be reconciled. They are used only for in-context inference; no model parameters are updated.

D.3 HIERARCHICAL-CONSOLIDATION PROMPT

When all partial graphs for a target case cannot fit within one request, they are recursively consolidated in batches. Intermediate generated backbones are then treated as inputs to the next consolidation level.

The following conceptual prompt applies the same reconstruction rules at each consolidation level:

> **Prompt: Hierarchical consolidation**
>
> You are given several candidate Bayesian-network graphs produced while consolidating partial descriptions of the same underlying system.
>
> Each candidate represents information originating from one or more source graphs.
>
> Combine the supplied candidates into one coherent, maximal plausible Bayesian-network backbone.
>
> Apply the same rules used for backbone reconstruction:
>
> - merge labels only when they identify the same variable;
> - reconcile state lists of matched variables;
> - preserve supported directed relationships;
> - retain justified probability information;
> - do not invent unsupported numerical probabilities;
> - recover omitted components only when supported by the supplied evidence;
> - provide concise justifications for inferred components;
> - ensure that all edge endpoints refer to included variables;
> - do not include self-loops or duplicate edges; and
> - return a valid directed acyclic graph conforming to the required JSON schema.
>
> Evidence represented by all supplied candidate graphs should be considered during consolidation.
>
> Do not assume access to the withheld reference backbone.

The prompt assumes that supplied probability tables are compatible. This is an assumption of the composition procedure, rather than a verified property of locally reparameterized subgraphs. The procedure stops when one candidate remains, with a maximum depth of 12. A case fails if a required request cannot fit, the next level does not reduce the item count, or source-identifier coverage is lost.

At each consolidation level, source identifiers are propagated separately by the evaluation procedure. Their union is required to equal the set of original source identifiers. This check establishes that every input graph participated in the hierarchy; it does not guarantee that every individual fact survived generation.

Requests are greedily batched under the configured serialized-input limit of 60,000 bytes, with a maximum output length of 3,000 tokens per request. The same three few-shot demonstrations accompany every generation request.

D.4 Output validation and evaluation

Generated outputs are subjected to JSON-schema and structural validation before evaluation. The validator requires unique node names and states, valid directed-edge endpoints, no self-loops or duplicate edges, and an acyclic directed graph.

When probability tables are included, they are additionally checked for dimensional compatibility, finite numerical values, values in $[0, 1]$, and normalization. Probability tables may be omitted when a relationship is inferred but its numerical values cannot be justified.

If a response contains nonnumeric, nonfinite, or out-of-range probability values, probability evaluation is skipped for that response while structural evaluation may continue. Other schema or DAG errors trigger the configured retry procedure.

Structural evaluation case-folds labels, replaces non-alphanumeric runs with spaces, and trims surrounding whitespace before exact matching. Nodes are evaluated as normalized label sets, states as node–state pairs, and edges as ordered endpoint pairs. CPD coverage measures the fraction of target numeric cells for which comparable predicted values are available and does not itself measure numerical agreement or calibration. Table 8 reports micro F1 from counts pooled across completed cases. The available summaries do not establish a shared evaluated subset across models. All models receive the same partial graphs, demonstrations, input and output limits, and validation rules, although provider-specific inference behavior can differ.

D.5 Selected backbone reconstructions

Table 11 reports two cases with complete structural recovery under deterministic matching. The cases were selected as the two highest-scoring completed examples in a separate source-text round-trip evaluation run. This selection favors successful examples and does not estimate representative backbone performance. Its text-comparison rubric differs from the graph-faithfulness rubric in Appendix E; those selection scores are not pooled with the latter evaluation.

The reference and reconstructed graphs are shown in Figures 11–14. The drawings display node labels, state cardinalities, and directed dependencies; Table 11 separately reports numerical coverage and accuracy from the same case records.

Metric	Climate and hantavirus	Ring modulation
Recovered nodes	9/9	13/13
Recovered node–state pairs	29/29	45/45
Recovered directed edges	10/10	13/13
Node / State / Edge F1	1.0000 / 1.0000 / 1.0000	1.0000 / 1.0000 / 1.0000
Comparable probability cells	96/96	0/165
Probability-cell coverage	1.0000	0.0000
Mean absolute probability error	1.43×10^{-5}	—
Maximum absolute probability error	4.55×10^{-5}	—
Complete probability columns	32	0
Mean column KL	3.17×10^{-9}	—

Table 11: Selected backbone reconstructions. Both recover all evaluated structural components. Numerical errors are reported for comparable probability entries; a dash indicates that no entries were available for comparison. These selected cases do not establish aggregate probability accuracy.

Climate change and hantavirus. Case `bn_00013` represents dependencies among climate, ecological conditions, rodent populations, hantavirus transmission, disease outcomes, and public-health consequences. The reconstruction recovers all nine nodes, 29 node–state pairs, and ten directed edges, without additional structural components. All 96 target probability cells have comparable predictions. Their mean absolute error is 1.43×10^{-5} and maximum absolute error is 4.55×10^{-5}; mean column KL is 3.17×10^{-9} over 32 complete columns. The supplied record therefore supports near-exact numerical recovery in this example as well as complete structural recovery. It does not establish calibration against real-world observations or a complete general BN parameterization.

Ring modulation. Case `bn_00031` represents signal-processing operations, circuit implementation, waveform and overtone content, sideband generation, harmonic relationships, and auditory output. The reconstruction recovers all 13 nodes, 45 node–state pairs, and 13 directed edges, without additional structural components. None of the 165 target probability cells has a comparable prediction, so numerical error is undefined rather than zero. Perfect structural scores can therefore coexist with zero probability coverage.

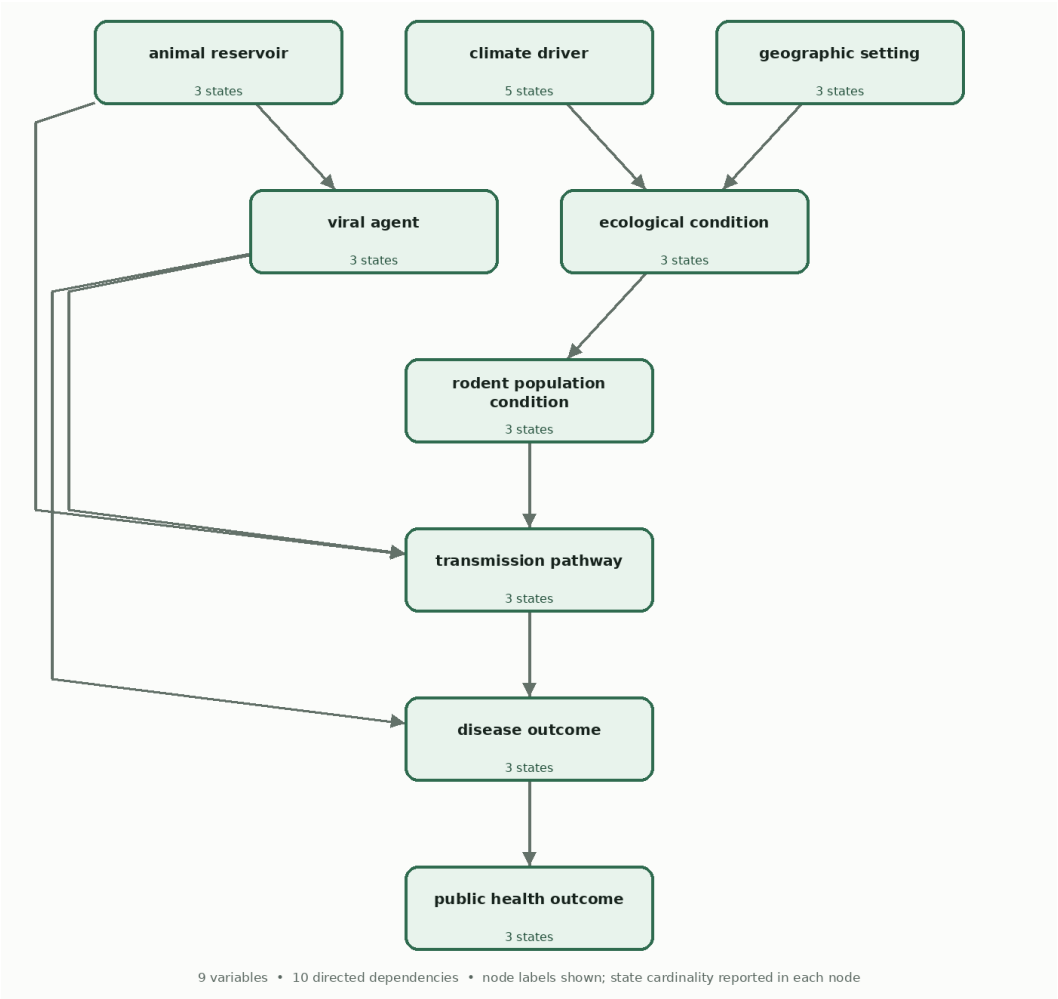


Figure 11: Reference backbone for `bn_00013`, with nine variables and ten directed dependencies. Each node reports its state cardinality; the drawings do not display probability values.

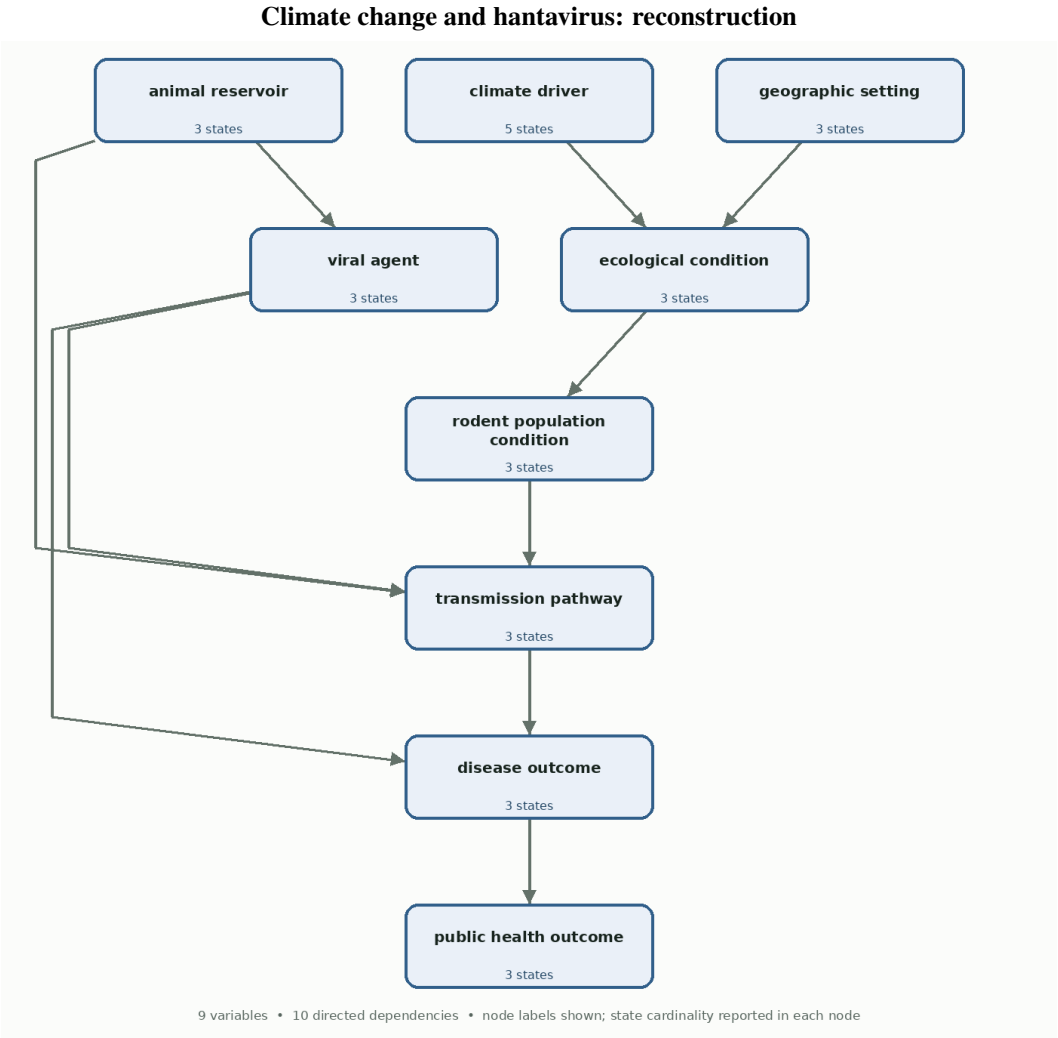


Figure 12: Reconstructed backbone for `bn_00013`. All nine nodes, 29 node–state pairs, and ten directed dependencies match the reference under deterministic label matching. All 96 target probability cells have comparable values, with mean absolute error 1.43×10^{-5}.

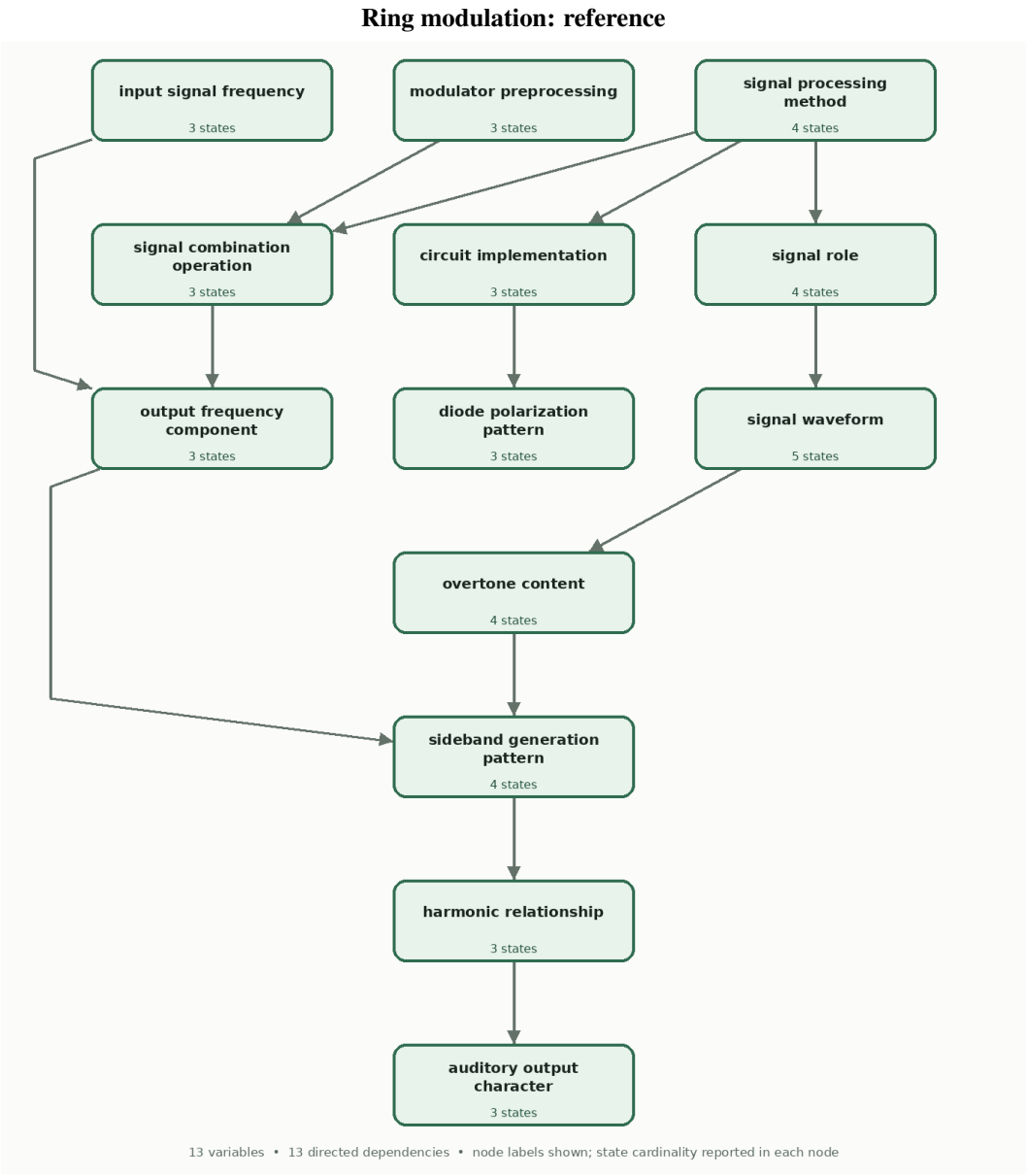


Figure 13: Reference backbone for `bn_00031`, with 13 variables, 45 node–state pairs, and 13 directed dependencies. State cardinalities appear within nodes.

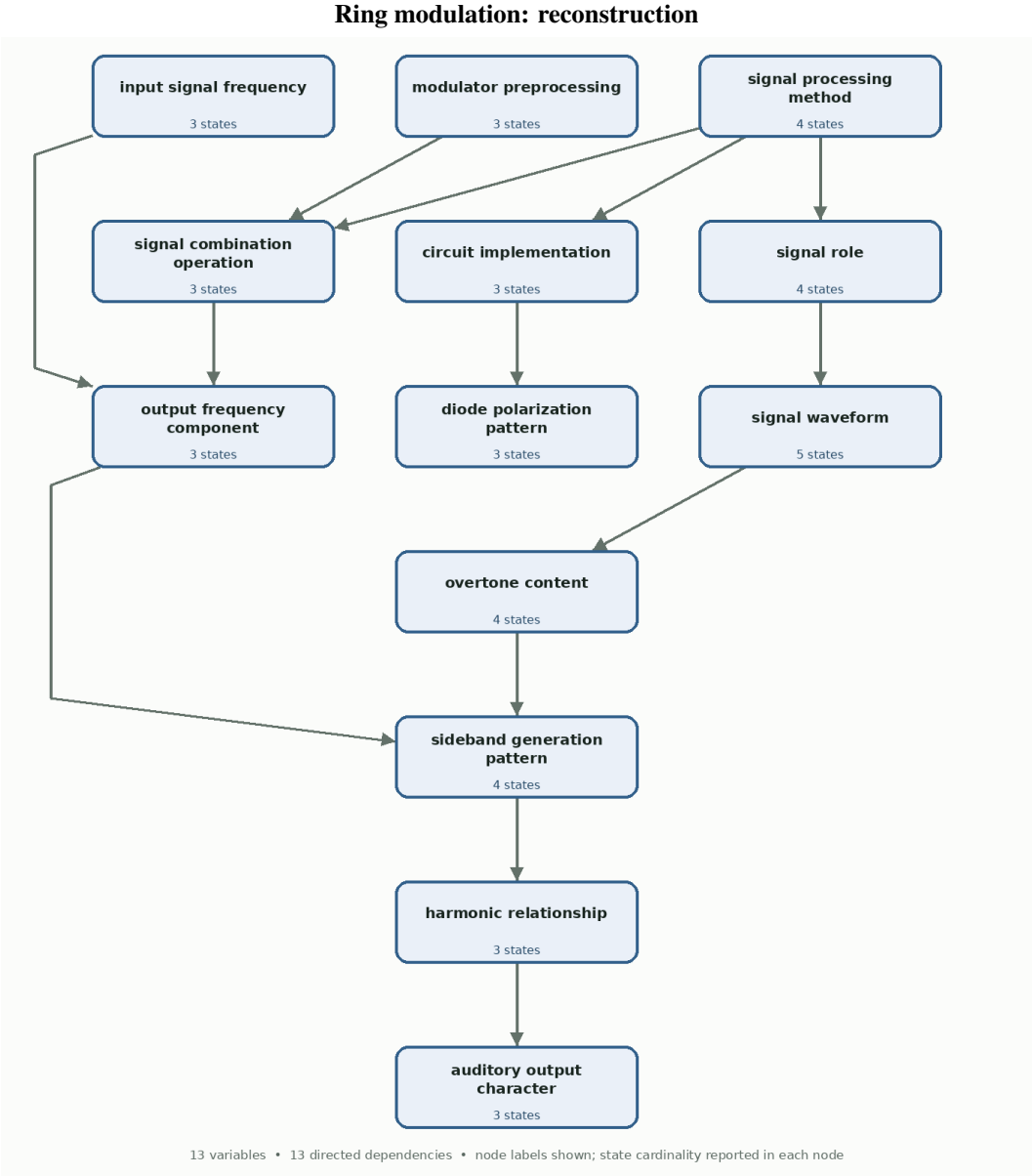


Figure 14: Reconstructed backbone for `bn_00031`. All evaluated nodes, states, and directed dependencies match the reference. None of the 165 target probability cells has a comparable prediction: the exact structural match does not imply probability recovery.

E SEMANTIC FAITHFULNESS EVALUATION

Generation and judging. GPT-5.6 converts each predicted backbone into article-style prose. The graph-to-text prompt includes the same three demonstration backbones used for consolidation, paired with example texts. The target article is withheld. GPT-6 Astra then judges the description against the untouched reference backbone. The judge does not receive the predicted backbone, structural scores, target article, or graph-to-text instructions. Generation and judging use separate calls.

Graph-to-text instructions. The generator is instructed to incorporate the concepts and meaningful states in the predicted backbone, express directed dependencies in connected prose, preserve their direction, and avoid causal interpretations unless supported. Numerical probabilities may be expressed only when supported by the available tables. The instructions prohibit invented events, mechanisms, statistics, or relationships introduced to make the prose more natural. They also disallow lists, JSON, graph terminology, and statements about missing information. This paragraph summarizes the instruction content rather than providing a serialized prompt.

Judge instructions and rubric. The judge checks whether claims about variables, states, and directed dependencies are supported by the reference graph. Semantic equivalence is allowed, but relationship types must be preserved when specified. The judge is instructed not to infer causation from a dependency unless the graph supports that interpretation. It assigns an integer score using the following rubric:

> **5**: Fully agrees; no materially unsupported claim.
> **4**: Mostly agrees; only a minor unsupported or imprecise claim.
> **3**: Partly agrees; contains both supported and unsupported claims.
> **2**: Mostly disagrees; substantial claims are unsupported or conflict with the graph.
> **1**: Does not agree; central claims are unsupported or contradict the graph.

Omitted graph information is not penalized. A description can receive a score of 5 without mentioning every node or relationship. The score therefore concerns support for expressed claims, rather than complete information preservation. Errors can arise during backbone consolidation, verbalization, or both; the evaluation does not isolate these sources.

Implementation. Graph-to-text generation uses low reasoning effort and a maximum output length of 7,000 tokens. Judging uses medium reasoning effort and a maximum of 800 tokens. Invalid responses are retried up to three times. Generated text must be non-empty prose rather than a list. Judge outputs must be JSON objects containing an integer score from 1 to 5 and a non-empty reason.

Scores and rationales. The 39 judged descriptions receive twenty scores of 3, thirteen of 4, and six of 5. The mean is 3.64 and the median is 3. Thus, 20 of 39 descriptions contain both supported and unsupported claims under the rubric. The rationales identify causal interpretations of probabilistic dependencies and temporal or mechanistic claims absent from the reference. One score-3 case preserves economic-relief concepts and numerical values but adds causal and chronological interpretations. A score-4 case preserves the principal concepts and dependency directions but interprets a $1.5°$C state as an explicit policy goal. These examples show why structural overlap alone is insufficient to establish the faithfulness of generated descriptions.

F HUMAN EVALUATION PROTOCOL

F.1 CASES, RATING ASSIGNMENT, AND AGGREGATION

The same three evaluators completed the subgraph and backbone tasks. Each task contained ten selected cases, with two cases from each of the five domains. All three evaluators rated every case and every displayed model output. There were no missing ratings.

The subgraph task presented four reconstructions per case, one from each model, giving 120 sets of criterion ratings in total and 30 ratings per model per criterion. The backbone task presented one reconstruction per case, giving 30 sets of criterion ratings. For each model and criterion, reported means average equally across cases and evaluators. Since every evaluator rated every output, averaging individual ratings gives the same mean as averaging the three ratings within each case and then averaging case means.

All quality ratings used a scale from 1 to 5. A score of 1 denoted poor or mostly incorrect reconstruction, 3 denoted mixed or partially correct reconstruction, and 5 denoted highly accurate reconstruction. Decimal ratings were allowed.

F.2 SUBGRAPH INTERFACE AND RUBRIC

The interface displayed a source passage followed by four generated reconstructions. These came from the SFT model, DUAL-BN with GRPO and $\lambda = 0$, DUAL-BN with GRPO and $\lambda = 0.05$, and GPT-6 Astra. Evaluators scored each reconstruction on five criteria.

- Node quality assessed whether the variables captured important concepts in the source without important omissions or unsupported additions.
- State quality assessed whether values and alternatives belonged to the appropriate variables.
- Edge quality assessed whether directed relationships were meaningful and their parent–child directions agreed with the source.
- Probability-table quality assessed whether conditional probabilities were plausible given the source description and displayed states.
- Overall quality assessed the reconstruction as a representation of the source text.

Evaluators also selected a preferred reconstruction for each case and could provide optional comments. Preference percentages summarize 30 evaluator–case choices and are rounded to whole numbers. Quality ratings and preference choices are reported separately in Appendix C.7.

F.3 BACKBONE INTERFACE AND RUBRIC

The backbone task assessed GPT-5.6 reconstructions composed from regularized DUAL-BN outputs. For each case, the interface showed the original source text, the reconstructed backbone, and a natural-language description generated from that backbone. Evaluators assigned four ratings on the same 1–5 scale.

- Node quality assessed whether the backbone represented the important concepts distributed across the source information.
- State quality assessed whether state spaces were appropriate for the recovered variables.
- Edge quality assessed whether directed relationships were meaningful and consistent with the source.
- Overall quality assessed whether the reconstructed backbone formed a coherent representation of the larger system.

The generated description provided an additional view of the reconstruction. The reported human scores concern backbone quality. They are separate from the LLM-based graph-to-text faithfulness scores in Appendix C.6. Human probability-table ratings were collected for subgraphs only.