From Implicit to Explicit Causal Models in BDI Agents: A Proof-of-Concept with Causal Discovery and Adaptive Policy

Meltem Gullusac[1][0000−0003−3765−4422], Baris Tekin Tezel[1][0000−0003−4873−7848], and Moharram Challenger[2,3][0000−0002−5436−6070]

[1] Department of Computer Science, Dokuz Eylül University, Izmir, Turkiye
meltem.gullusac@deu.edu.tr, baris.tezel@deu.edu.tr
[2] Department of Computer Science, University of Antwerp, Antwerp, Belgium
moharram.challenger@uantwerpen.be
[3] AnSyMo/CoSys Core-lab, Flanders Make Strategic Research Centre, Leuven, Belgium

Abstract. Belief-Desire-Intention (BDI) agents encode causal assumptions implicitly within their plan libraries and belief-update rules, without formally representing or reasoning over causal structure. This limits their ability to detect confounding, distinguish correlation from causation, and adapt to environments with latent variables. We present CausalBDI, an architectural pattern that augments Jason/AgentSpeak agents with an explicit structural causal model (SCM) maintained by an external causal inference server. The agent progresses through an epistemic lifecycle: (1) a *naive phase* with epsilon-greedy exploration to collect randomised-policy data, (2) *causal discovery* via Fast Causal Inference (FCI) on data from random action selection to detect latent confounders, (3) an *epistemic transition* where beliefs shift from correlational to causal, and (4) a *causal policy phase* using proxy-stratified estimation, sensitivity analysis, and Thompson Sampling. We evaluate the architecture on a navigation scenario where a proxy variable (yellow warning light) misleads naive agents into suboptimal behaviour: over 30 independent runs, the causally-aware agent reduces accident rate by 25% compared to the naive baseline ($p < 0.001$), closing 42% of the gap to an oracle with direct access to the latent confounder. Our work provides a concrete proof-of-concept for integrating causal learning and reasoning into the BDI reasoning cycle, bridging the gap between agent programming and causal inference.

Keywords: BDI agents · causal reasoning · structural causal models · causal discovery · Jason/AgentSpeak · Thompson Sampling · latent confounders

1 Introduction

The Belief-Desire-Intention (BDI) architecture [15] provides an elegant computational model for rational agents, where beliefs encode the agent's informational state, desires represent objectives, and intentions commit the agent to courses of action via plan execution. Platforms such as Jason [4] and JaCaMo [3] have made BDI programming accessible through the AgentSpeak language, enabling agents that operate reactively and proactively in complex environments.

However, as recently emphasised by the CLaRAMAS workshop organisers [1], the causal knowledge underpinning BDI agent behaviour remains *implicit*. A BDI plan library constitutes an informal causal model: the developer encodes assumptions about which actions lead to which outcomes, but no formal causal structure is made available for the agent to inspect, refine, or reason over. This implicit encoding has several consequences: (i) the agent cannot distinguish spurious correlation from genuine causation, (ii) it cannot detect latent confounders that distort observed associations, and (iii) it cannot perform interventional or counterfactual reasoning about its own actions.

These limitations become acute in environments with *unobserved confounders*—latent variables that simultaneously influence both the agent's observations and its outcomes. In such settings, a naive agent that conditions only on observed variables may adopt a systematically biased policy, a well-documented phenomenon in causal inference [14,10].

We present CausalBDI, an architecture that bridges this gap by augmenting a Jason BDI agent with an *explicit structural causal model* (SCM). The agent actively discovers the causal structure of its environment, estimates causal effects with appropriate uncertainty quantification, and adapts its policy accordingly. Rather than proposing a fully general causal reasoning framework, we focus on a concrete proof-of-concept that demonstrates the feasibility and value of making causal models explicit within BDI agents. Our contributions:

1. A structured **epistemic lifecycle** for causal BDI agents, comprising naive exploration, causal discovery, epistemic transition, and causal exploitation phases (Sect. 3).
2. A **concrete implementation** in Jason/JaCaMo with an external Python-based causal inference server, demonstrating that explicit causal reasoning can be integrated into BDI platforms through a loosely coupled service-oriented design that preserves the standard BDI reasoning cycle. (Sect. 6).
3. Attention to **methodological soundness**—randomised-policy data for valid FCI, proxy-stratified estimation with honest uncertainty quantification, sensitivity analysis, and Thompson Sampling for causal bandits—while explicitly acknowledging what is and is not identified (Sect. 4).
4. A **demonstration scenario** with latent confounding where the naive agent is systematically deceived, while the causally-aware agent is designed to identify the proxy structure (Sect. 5).
5. A **quantitative evaluation** comparing naive, causally-aware, and oracle agents over 30 independent simulation runs, demonstrating statistically sig-

nificant improvement in accident rate (the proportion of simulation steps in which an accident occurs) after causal discovery (Sect. 7).

Notation. We use the following abbreviations throughout: BDI (Belief-Desire-Intention), SCM (Structural Causal Model), DAG (Directed Acyclic Graph), PAG (Partial Ancestral Graph), FCI (Fast Causal Inference), HPm (modified Halpern-Pearl definition of actual causality), IPW (Inverse Probability Weighting), KQML (Knowledge Query and Manipulation Language). Scenario variables: C = Cleaning, S = Slippery (latent), Y = Yellow (observed proxy), A = Accident, U = agent's chosen action.

2 Background and Related Work

Structural Causal Models. Pearl's SCM framework [14] formalises causal relationships through directed acyclic graphs (DAGs) and structural equations. A key insight is the distinction between *observational* distributions $P(Y \mid X)$, *interventional* distributions $P(Y \mid \mathrm{do}(X))$, and *counterfactual* queries. In the presence of unobserved confounders, observational conditioning does not generally yield valid causal estimates—precisely the problem that naive agents face when they treat correlational beliefs as causal.

Causal Discovery. Constraint-based algorithms such as FCI (Fast Causal Inference) [21] can discover causal structure from data, including the presence of latent common causes. Unlike the simpler PC algorithm, FCI produces *partial ancestral graphs* (PAGs) that correctly represent ambiguity due to unobserved variables. A critical requirement is that the input data satisfy the algorithm's assumptions; in an agent setting, this means that observations used for discovery should come from randomised (interventional) data to avoid confounding the discovery process itself.

Causal Bandits and Sensitivity Analysis. The connection between causality and sequential decision-making has been formalised in the causal bandit literature [12,2], where agents choose actions in settings described by causal models. Proxy variable approaches [13] and sensitivity analysis [16] provide tools for reasoning under unmeasured confounding. However, these methods have not been integrated into practical BDI agent platforms.

BDI Agents and Implicit Causality. BDI architectures [15,4] represent a mature paradigm for cognitive agent programming. A plan $+e : c \leftarrow a_1; a_2; \ldots$ implicitly assumes that executing $a_1, a_2, \ldots$ under context c will causally bring about the desired outcome. Yet this causal model is neither explicit nor revisable by the agent. Recent surveys on BDI architectures [6] note the growth of learning-based extensions (e.g., via reinforcement learning), but these encode causal knowledge implicitly in policies or value functions. Our work addresses this gap directly by providing BDI agents with an explicit, inspectable, and revisable causal model.

Causality in Multi-Agent Systems. A growing body of work investigates the formal relationship between causality and multi-agent systems. In one line of inquiry, Gladyshev et al. [7] propose Dynamic Causality, an alternative to the modified Halpern-Pearl HP^m definition of actual causality [9] based on the order in which variable values are computed given the dependency structure of a causal model; the key insight is that an intervention on a candidate cause may not prevent the effect from occurring, but can delay the computational step at which the effect is determined — yielding more intuitive causal ascriptions in cases where HP^m fails (such as the Switches problem [9]) and lower verification complexity for unary causes. In a related line, the same group [8] proposes a novel temporal interpretation of structural equation models in which SEMs are viewed as mechanisms transforming time series of exogenous values into time series of endogenous values, combining counterfactual causal reasoning with linear-time temporal logic (LTL) to obtain the logic CPLTL; a notable feature is that non-recursive (cyclic) models receive a natural temporal interpretation, avoiding the technical difficulties they pose in static settings. In a parallel line, Kerkhove et al. [11] introduce a systematic method for constructing concurrent game structures from structural causal models, enabling analysis of how agents' strategic decisions produce causal effects and connecting causal and strategic notions of responsibility. On the information-theoretic side, Simoes et al. [20,18] develop causal entropy and causal information gain to measure how much control an intervention provides over outcomes, and recently characterise the minimal search space for conditional causal bandits [19].

These contributions establish formal foundations for causal reasoning *given* a structural causal model. Our work addresses a complementary question at a more practical level: given a set of known variables with pre-specified roles (treatment, outcome, covariate), can a BDI agent use data-driven methods to determine the *nature* of the causal relationships between them—in particular, whether an observed variable is a direct cause or a proxy for a latent confounder? While this falls short of open-ended causal discovery over arbitrary variable sets, it demonstrates that even limited structural refinement via FCI can meaningfully change an agent's epistemic state and yield measurable policy improvements—suggesting a practical entry point for integrating causal learning into BDI architectures.

3 The CausalBDI Architecture

3.1 Overview

Our architecture introduces an *epistemic lifecycle* that mirrors how a rational agent would progress from ignorance to causal understanding. The agent maintains a meta-belief `belief_structure` that tracks its current epistemic state—`naive` (correlational) or `causal_confounding` (post-discovery). The lifecycle comprises four phases:

1. **Naive Exploration:** The agent acts according to correlational beliefs with ε-greedy exploration ($\varepsilon = 0.4$), ensuring $\sim$40% of observations come from

uniformly random action selection. Each observation is tagged with an `is_random` flag. Discovery is attempted every 25 steps, but the server requires $n_{\text{random}} \geq 50$; with $\varepsilon = 0.4$, approximately 10 randomised observations accumulate per interval, so several early discovery attempts return `insufficient` before the threshold is reached (typically around step 125-150).

2. **Causal Discovery:** Periodically, the agent invokes FCI on *only the randomised observations*. This is methodologically critical: the agent's own policy creates dependencies between observed variables and actions; using all data would introduce spurious edges.
3. **Epistemic Transition:** When FCI reveals that an observed variable is a *proxy* for a latent confounder (rather than a direct cause), the agent updates its belief structure from `naive` to `causal_confounding` and runs sensitivity analysis to calibrate confidence.
4. **Causal Policy:** The agent switches to Thompson Sampling [22,5] with proxy-stratified posteriors, replacing the naive correlational policy. Thompson Sampling selects actions by sampling from posterior distributions over expected outcomes, naturally balancing exploration and exploitation without requiring explicit confidence-bound tuning. Confidence level (high/medium/low) is set by the Rosenbaum sensitivity tipping point Γ^*.

The transition from exploration to exploitation is governed by a *phase readiness* check: the server verifies both sufficient randomised data ($n_{\text{random}} \geq 50$) and adequate overlap (at least 8 observations per action per Y-stratum, i.e., per value of Y) before signalling readiness for causal exploitation.

Figure 1 illustrates the overall architecture.

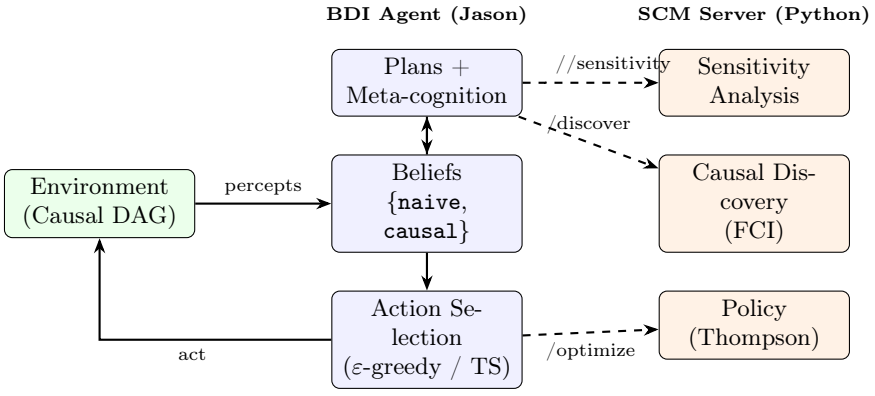

Fig. 1. The CausalBDI architecture. The BDI agent (left) interacts with the environment and queries an external causal inference server (right) via HTTP/JSON. Solid arrows: standard BDI perception–action loop; dashed arrows: causal inference queries.

3.2 Epistemic Meta-Beliefs

A key design principle is that the agent's *epistemic state* is itself a belief, enabling introspective reasoning:

```
belief_structure(naive).          // or: causal_confounding
belief_latent_danger(0.0).        // P(LatentDanger | Proxy)
belief_confidence(low).           // low / medium / high
```

The transition from `naive` to `causal_confounding` constitutes an *epistemic revolution*: the agent revises not merely individual beliefs, but its entire model of how observations relate to outcomes. In the naive phase, the agent interprets $\mathit{Yellow} = 1$ as high danger ($P = 0.9$); after the epistemic transition, it recognises Yellow as a proxy with attenuated evidential value ($P \in [0.55, 0.65]$ depending on confidence).

4 Methodological Foundations

4.1 Valid Causal Discovery with Interventional Data

A naïve application of constraint-based causal discovery to agent-generated data is methodologically flawed: the agent's own policy creates a dependency $\mathit{Yellow} \to \mathit{Action}$ that is an artefact of the decision process, not of the environment's causal structure. To address this, our architecture ensures that FCI receives *only data collected under random action selection* (the ε-greedy exploration steps). Under random action selection, $\mathit{Action} \perp\!\!\!\perp \mathit{Yellow}$, removing the policy-induced spurious association and allowing FCI to detect the true latent structure. The server enforces a minimum of $n_{\text{random}} \geq 50$ before attempting discovery.

4.2 Proxy-Stratified Estimation

Stratification. Stratification is a general statistical technique for estimating conditional effects: the data is partitioned into subgroups (strata) defined by the value of one or more covariates, and effect estimates are computed within each stratum. In our scenario, we stratify by the proxy variable Y, yielding two Y-strata (corresponding to Y=0 and Y=1). The Y-stratified estimator below thus instantiates a general method to our specific causal structure; references to "Y-stratum" throughout the paper denote this scenario-specific application of the general concept.

In our scenario, the true DAG is *Cleaning* (C) $\to$ { *Yellow* (Y), *Slippery* (S)}, with both $S \to \mathit{Accident}$ (A) and $U \to A$, where $U \in \{\mathit{fast}, \mathit{slow}\}$ denotes the agent's chosen action (the treatment variable). Since S is unobserved, Y is a proxy for C (the common ancestor of the proxy and the confounder), not for S itself. Standard adjustment (Inverse Probability Weighting (IPW) or stratification) on Y does *not* suffice to identify the causal effect, because Y is a proxy for C—not for S itself—and does not block the confounding association between U and A that arises when the agent's policy conditions on Y (creating the path

$S \leftarrow C \rightarrow Y \rightarrow_{\text{policy}} U$). Consequently, the causal effect $P(A{=}1 \mid \text{do}(U{=}a))$ is *not point-identified* from observed data. Instead, we compute an *observational* proxy-stratified estimate:

$$\hat{R}_{\text{str}}(a) \;=\; \sum_{y \in \{0,1\}} P(Y{=}y) \cdot P(A{=}1 \mid U{=}a,\, Y{=}y) \tag{1}$$

This is *not* the interventional quantity $P(A{=}1 \mid \text{do}(U{=}a))$, but an approximation whose bias depends on how well Y captures the confounding. We therefore report stratum-specific risk differences as informative *bounds*: if the sign of $\hat{R}_{\text{str}}(\textit{fast}) - \hat{R}_{\text{str}}(\textit{slow})$ agrees across both Y-strata , this provides evidence for the direction of the causal effect, even though the magnitude is not identified.

4.3 Rosenbaum Sensitivity Analysis

To quantify robustness to unmeasured confounding, we employ a sensitivity analysis inspired by Rosenbaum's framework [16]. The core idea is that an unmeasured confounder could shift the true risk rates away from the observed ones. For each sensitivity parameter $\Gamma \geq 1$, we compute bounds on the risk difference by shifting the observed rates within a Y-stratum by $\delta(\Gamma) = (\Gamma - 1)/(\Gamma + 1)$:

$$RD_{\text{lower}}(\Gamma) \;=\; \Big(\hat{r}_{\text{fast}} - \delta(\Gamma)\Big) - \Big(\hat{r}_{\text{slow}} + \delta(\Gamma)\Big) \tag{2}$$

Note that this is a *simplified* heuristic: Rosenbaum's original method operates on matched pairs and bounds the Wilcoxon signed-rank statistic, whereas our formulation directly shifts risk rates by a quantity derived from the propensity odds ratio. The simplification sacrifices formal guarantees but retains the key insight: $\Gamma = 1$ recovers the unadjusted estimate, and the *tipping point* Γ^* where the bound interval first contains zero provides a useful measure of fragility. Our implementation maps: $\Gamma^* > 3.0 \Rightarrow$ high confidence, $\Gamma^* > 1.5 \Rightarrow$ medium, $\Gamma^* \leq 1.5 \Rightarrow$ low. This value directly updates the agent's `belief_confidence`.

4.4 Thompson Sampling for Causal Bandits

In the causal policy phase, we use Thompson Sampling [22,5] with Y-stratified Beta posteriors. For each action a in Y-stratum y, the posterior on safety probability is:

$$\theta_{a,y} \;\sim\; \text{Beta}(\alpha_{a,y} + 1,\; \beta_{a,y} + 1) \tag{3}$$

where $\alpha_{a,y}$ counts safe outcomes and $\beta_{a,y}$ counts accidents. The agent samples from each posterior and selects the action with higher sampled safety, naturally balancing exploration and exploitation. When sufficient data is available, it switches to posterior mean exploitation.

5 Demonstration Scenario

5.1 Environment: The Wet Floor Navigation Scenario

We demonstrate CausalBDI on a navigation scenario with the causal structure depicted in Fig. 2. The variable *Yellow* represents a yellow warning light that indicates a recently-cleaned (and therefore potentially slippery) area. The agent observes *Yellow* directly but cannot perceive whether the floor is actually slippery—it must reason about the latent *Slippery* state from the observable proxy.

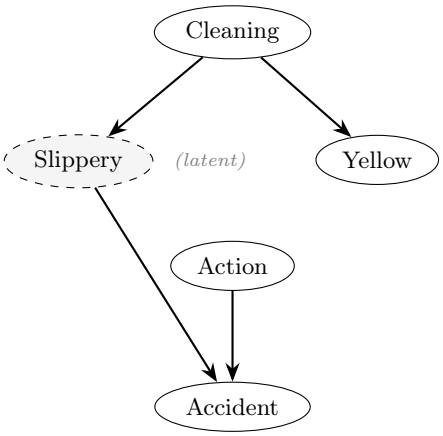

Fig. 2. Causal DAG. *Cleaning* causes both *Slippery* (latent, dashed) and *Yellow* (observed). Yellow is a proxy for Cleaning—the common ancestor—not for the confounder Slippery itself. The agent observes only Yellow but must reason about *Action*→*Accident*, confounded by the latent Slippery.

The data-generating process, implemented as a second Jason agent (`env.asl`) that communicates percepts to the cognitive agent via KQML `tell`/`untell` messages (rather than a JaCaMo environment artifact), is specified by conditional probability tables:

- $P(\mathit{Cleaning}{=}1) = 0.30$
- $P(\mathit{Slippery}{=}1 \mid C{=}1) = 0.90$; $P(\mathit{Slippery}{=}1 \mid C{=}0) = 0.05$
- $P(\mathit{Yellow}{=}1 \mid C{=}1) = 0.80$; $P(\mathit{Yellow}{=}1 \mid C{=}0) = 0.10$
- $P(\mathit{Accident} \mid S, A)$: see Table 1

A naive agent observing $\mathit{Yellow}{=}1$ and high accident rates may erroneously conclude that Yellow *causes* accidents, or that its mere presence necessitates slowing down. The true structure is more subtle: Yellow is a proxy for Cleaning (the common cause), which also causes Slippery—the actual latent confounder whose interaction with Action determines accident risk.

Table 1. Accident probabilities $P(Accident{=}1 \mid Slippery, Action)$.

	Action = fast	**Action = slow**
Slippery = 1	0.70	0.10
Slippery = 0	0.01	0.005

5.2 Agent Behaviour Across Phases

Naive Phase. The agent begins with `belief_structure(naive)` and interprets Yellow as a direct danger signal. With $\varepsilon = 0.4$ exploration, it collects randomised observations (tagged `is_random(true)`) alongside policy-driven ones.

Discovery and Transition. Every 25 steps, the agent invokes `scm.run_discovery(Status)`, which calls FCI on randomised data only. The server interprets the resulting PAG edge marks between Yellow (column 0), Action (column 1), and Accident (column 2) according to FCI's encoding (-1 = tail, 1 = arrowhead, 2 = circle, 0 = no edge):

(i) if the Yellow–Accident edge is absent (d-separated) or bidirected ($\leftrightarrow$, marks $(1,1)$), the status is `proxy`—indicating a latent common cause;
(ii) if a circle mark ($\circ\!\!\rightarrow$, marks $(2,1)$) appears on Y–A with $Y \not\perp\!\!\!\perp$ Action confirmed by the absence of a Y–U edge (guaranteed by randomisation), this also signals proxy structure with high confidence;
(iii) if FCI returns $Y \rightarrow A$ (marks $(-1,1)$, direct arrow), the agent's naive model remains valid (`direct_cause`) and no epistemic transition occurs;
(iv) if a circle mark appears on Y–A but a Y–U edge is also present (suggesting imperfect randomisation), the result is `ambiguous_latent` and the agent transitions with caution.

```
if (Status == proxy) {
    -+belief_structure(causal_confounding);
    scm.set_epoch(1);       // signal new epistemic phase
    !assess_confidence;      // run sensitivity analysis
};
if (Status == ambiguous_latent) {
    -+belief_structure(causal_confounding);
    scm.set_epoch(1);
    !assess_confidence;
};
if (Status == direct_cause) {
    .print("FCI: Yellow -> Accident (direct). Naive model valid.");
};
```

Causal Phase. Post-transition, the agent's latent danger estimates become nuanced: $P(Danger \mid Y{=}1) \in [0.55, 0.65]$ (vs. naive 0.9), reflecting the proxy understanding. Action selection delegates to Thompson Sampling via the causal server.

6 Implementation

The system consists of three loosely coupled components communicating via HTTP/JSON:

Environment Agent (`env.asl`). A second Jason agent (not a JaCaMo environment artifact) implementing the causal DAG as a generative model. It produces percepts (Yellow, Accident, position, fuel) via KQML `tell`/`untell` messaging to the cognitive agent. The latent variable Slippery is never communicated—the cognitive agent must infer its influence.

Cognitive Agent (`a1.asl`). A Jason BDI agent whose plan library implements the epistemic lifecycle. Custom Java internal actions in the `scm` package provide a clean interface:

- `scm.record_observation(Y, Act, Acc, IsRandom)` — records data with randomisation flag
- `scm.run_discovery(Status)` — FCI returning structure classification
- `scm.test_intervention(RiskFast, RiskSlow)` — proxy-stratified estimation
- `scm.run_sensitivity(TippingGamma)` — Rosenbaum analysis returning Γ^*
- `scm.get_optimal_action(Y, Explore, Best)` — Thompson Sampling query

Causal Inference Server (`scm_server.py`). A Python HTTP server using `causal-learn` [23] for FCI, NumPy for estimation and Thompson Sampling, and custom Rosenbaum bounds. The server maintains observation history with metadata (randomisation flag, epoch) and filters data appropriately for each task. The loose coupling makes it straightforward to replace the server with DoWhy [17] or to deploy it as a shared service for multiple agents.

7 Experimental Evaluation

To assess whether the epistemic lifecycle yields measurable behavioural improvement, we conducted a quantitative evaluation comparing three agent types over 30 independent simulation runs.

7.1 Setup

The experiment replicates the agent's decision logic and the causal inference pipeline in a standalone Python simulation for reproducibility and statistical evaluation. The environment uses the same causal DAG and probability tables described in Sect. 5, with parameters: goal distance = 500, initial fuel = 1000, $\varepsilon = 0.4$, and FCI minimum sample size $n_{\text{random}} = 50$. Each step, the agent chooses *fast* (+3 progress, −1 fuel) or *slow* (+1 progress, −1 fuel); accidents yield zero progress and a fuel penalty of 5. The simulation ends when the agent reaches the goal or exhausts fuel.

Three agent types are compared:

- **Naive:** Uses only correlational beliefs throughout (*Yellow*=1 ⇒ *slow*, *Yellow*=0 ⇒ *fast*). No causal discovery is performed.
- **CausalBDI:** Implements the full epistemic lifecycle: ε-greedy exploration, FCI-based discovery, epistemic transition, and Thompson Sampling with proxy-stratified posteriors.
- **Oracle:** Observes the latent variable *Slippery* directly (*Slippery*=1 ⇒ *slow*, *Slippery*=0 ⇒ *fast*). This represents a theoretical upper bound, as the confounder is unobservable in practice.

7.2 Results

Table 2 summarises the results across 30 independent runs per agent type.

Table 2. Agent comparison over 30 independent simulation runs. Values reported as mean ± std. The CausalBDI agent achieves a statistically significant reduction in accident rate compared to the naive baseline (Welch's $t = 5.31$, $p < 0.001$).

Metric	Naive	CausalBDI	Oracle
Success Rate	30/30	30/30	30/30
Steps to Goal	233 ± 7	349 ± 58	214 ± 6
Accident Rate	0.091 ± 0.016	$\mathbf{0.068 \pm 0.016}$	0.037 ± 0.012
Fuel Remaining	682 ± 23	559 ± 64	754 ± 15
Discovery Rate	—	30/30 (100%)	—
Discovery Step	—	139 ± 18	—

The CausalBDI agent achieves an accident rate of 0.068 ± 0.016, representing a 25% reduction compared to the naive agent's 0.091 ± 0.016. This difference is statistically significant (Welch's t-test: $t = 5.31$, $p < 0.001$). FCI correctly identifies the proxy structure in all 30 runs, triggering the epistemic transition at step 139 ± 18 on average.

To contextualise the improvement: the Oracle agent achieves an accident rate of 0.037 ± 0.012 with perfect knowledge of the latent confounder. The gap between naive and oracle is 0.054; CausalBDI closes 42% of this gap (0.023/0.054) using only the observed proxy variable and data-driven causal discovery—without ever observing the true confounder.

The increased step count for CausalBDI (349 vs. 233) and reduced fuel remaining (559 vs. 682) reflect the exploration cost of the ε-greedy phase: approximately 40% of actions are randomised during the naive phase, both slowing progress and inducing additional accidents (with their fuel penalties) before causal structure is identified. The Oracle's lead on these efficiency metrics (214 steps, 754 fuel) reflects its access to the latent confounder, which removes the need for exploration entirely—an unattainable advantage in practice. This exploration–exploitation tradeoff is inherent to the architecture; the agent invests in data collection to enable valid causal discovery, which subsequently yields a

safer policy. We note that the Rosenbaum sensitivity tipping point $\Gamma^* = 1.25$ indicates low robustness to unmeasured confounding, consistent with Yellow being an imperfect proxy—an honest diagnostic that the agent itself produces.

8 Discussion

Explicit vs. Implicit Causal Models. Our architecture makes the causal model *explicit* and *revisable*—a fundamental departure from standard BDI practice. The agent distinguishes proxies from causes and quantifies sensitivity to unmeasured confounding, moving from Pearl's associational level ($P(Y \mid X)$) toward *recognising the gap* between observational and interventional quantities ($P(Y \mid do(X))$) [14] —even where the latter is not point-identified.

Causal Discovery and BDI Plan Revision. Our current implementation partially leverages Jason's plan selection mechanism—for instance, `+!infer_latent_state` is defined with two context-differentiated plans selected based on `belief_structure` but the plan library itself remains static throughout execution: causal discovery updates a belief flag rather than revising or generating plans at runtime. A more idiomatic BDI integration would involve **dynamic plan library revision**, moving causal reasoning from a belief-level mode switch to the core of the deliberation cycle. This deeper integration can be achieved through two primary mechanisms:

- **Context Revision:** Instead of internal `if-else` logic, the agent could automatically update plan context conditions to reflect discovered structures. For instance, plans relying on variables identified as proxies could have their applicability conditions attenuated or conjoined with the agent's calibrated confidence (e.g., `yellow(1) & belief_confidence(high)`).
- **Plan Generation:** Upon identifying latent confounding, the agent could synthesize entirely new plans—such as instrumental exploration strategies—that would be irrational under a naive model but become valuable for refining causal estimates.

By utilizing Jason's `.add_plan` and `.remove_plan` primitives, causal discovery moves beyond simple action–value estimation; it informs the agent's practical reasoning by directly determining which plans are considered and which intentions are formed. We leave the full realization of this self-revising architecture, including formal guarantees on plan library consistency, as a significant direction for future work.

Methodological Soundness. A key contribution is the attention to inferential validity, even where the implementation falls short of ideal. Causal discovery uses only randomised-action data, avoiding the common mistake of applying FCI to confounded agent-generated data. The proxy-stratified estimator honestly reports bounds rather than claiming point identification, and the sensitivity

analysis—though a simplified heuristic—provides the agent with a concrete measure of how fragile its conclusions are. We note that the Thompson Sampling posteriors currently use all data (including policy-driven observations), which introduces potential bias; addressing this is important future work.

Assumptions and Required Prior Knowledge. The CausalBDI architecture, as instantiated in this proof-of-concept, relies on a set of explicit assumptions that should be made transparent before practical deployment. We organise them into three categories. *Structural assumptions:* The agent is assumed to know *a priori* (i) the set of relevant observable variables (Y, U, A in our scenario), (ii) the role of each variable (treatment, outcome, candidate proxy/covariate), and (iii) the existence, if latent confounding is present, of an observed proxy variable that captures partial information about it. FCI's role is therefore restricted to determining the *nature* of the relationship between known variables (direct cause, proxy, or latent-mediated), not open-ended structure discovery over arbitrary variable sets. *Statistical assumptions:* Causal discovery via FCI requires (i) faithfulness (independencies in the data correspond to d-separation in the true graph), (ii) sufficient sample size ($n_{\text{random}} \geq 50$ in our setup, with adequate overlap across (action, Y-value) cells), and (iii) data collected under randomised action selection so that $U \perp\!\!\!\perp Y$ holds by construction. The Thompson Sampling and proxy-stratified estimation phases additionally assume binary variables and stationarity of the underlying causal mechanisms. *Operational assumptions:* The agent must be permitted to perform randomised interventions during the naive exploration phase. In safety-critical or cost-sensitive applications this assumption may fail; hybrid schemes that combine limited interventional with observational data would be required. These assumptions are realistic for semi-controlled monitoring tasks but become limiting in open-ended environments.

Limitations and Future Work. The current implementation has several limitations that should be noted. First, while causal *discovery* uses only randomised-action data, the Thompson Sampling posteriors and proxy-stratified estimates are computed from *all* observations, including policy-driven ones. This means these estimates may carry residual confounding bias from the agent's own action-selection policy—an important caveat. Separating estimation by epoch or using only randomised data for posterior updates would improve inferential validity at the cost of sample efficiency. Second, the implementation restricts variables to binary and assumes a fixed DAG topology. Third, the architecture assumes the agent can perform randomised interventions, which may not always be feasible. Fourth, FCI's statistical power with 50 binary observations is limited, and discovery results should be interpreted cautiously. Fifth, the PAG interpretation logic in the current implementation is scenario-specific: the server knows the column ordering and interprets edge patterns between pre-identified variables. The agent does not perform fully generic structure discovery over an arbitrary variable set. Rather, FCI's role is to determine the nature of the relationship (direct cause, proxy, latent confounder) between variables whose roles (treatment, outcome, covariate) are already known to the agent—a form of structural refine-

ment rather than open-ended discovery. A generic PAG interpreter that could operate over arbitrary variable sets and automatically identify proxy structures represents important future work toward a truly general-purpose causal BDI architecture.

Broader Implications. Causal inference tools map naturally onto BDI belief-revision: discovery revises structural beliefs, estimation updates risk assessments, and sensitivity analysis calibrates confidence. Combined with the plan revision mechanisms discussed above, this suggests a broader programme where causal models are first-class citizens of the agent's mental state—not only informing *what* the agent believes, but also *how* it deliberates and *which* plans it considers applicable. Multi-agent extensions—where agents share causal knowledge or reason about each other's causal models—represent a compelling future direction.

9 Conclusion

We have presented CausalBDI, an architecture that augments Jason BDI agents with explicit structural causal models. By progressing through a principled epistemic lifecycle—from naive correlational beliefs, through causal discovery and sensitivity analysis, to causally-informed adaptive policy—the agent achieves a qualitatively different and more robust form of reasoning compared to standard BDI agents. Our experimental evaluation over 30 independent simulation runs demonstrates that the causally-aware agent reduces accident rate by 25% compared to the naive baseline, closing 42% of the gap to an oracle agent with perfect knowledge of the latent confounder. The full implementation, including Jason agent code, causal inference server, and evaluation scripts, is available at https://github.com/meltemino/CausalBDI.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. CLaRAMAS Workshop: 1st Int. Workshop on Causal Learning and Reasoning in Agents and Multiagent Systems. https://claramas-workshop.github.io/claramas2026/ (2026), last accessed 2026/02/24
2. Bareinboim, E., Forney, A., Pearl, J.: Bandits with unobserved confounders: A causal approach. Advances in neural information processing systems **28** (2015)
3. Boissier, O., Bordini, R.H., Hübner, J.F., Ricci, A., Santi, A.: Multi-agent oriented programming with jacamo. Science of Computer Programming **78**(6), 747–761 (2013)
4. Bordini, R.H., Hübner, J.F., Wooldridge, M.J., Wooldridge, M.J.: Programming multi-agent systems in AgentSpeak using Jason, vol. 8. Wiley Online Library (2007)
5. Daniel, J.R., Benjamin, V.R., Abbas, K., Ian, O., Zheng, W.: A tutorial on thompson sampling. Foundations and Trends® in Machine Learning **11**(1), 1–99 (2018)

6. De Silva, L., Meneguzzi, F.R., Logan, B.: Bdi agent architectures: A survey. In: Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI), 2020, Japão. (2020)
7. Gladyshev, M., Alechina, N., Dastani, M., Doder, D., Logan, B.: Dynamic causality. In: ECAI 2023-26th European Conference on Artificial Intelligence, including 12th Conference on Prestigious Applications of Intelligent Systems, PAIS 2023-Proceedings. pp. 867–874. IOS Press (2023)
8. Gladyshev, M., Alechina, N., Dastani, M., Doder, D., Logan, B.: Temporal causal reasoning with (non-recursive) structural equation models. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 39, pp. 14949–14957 (2025)
9. Halpern, J.Y.: Actual causality. MiT Press (2016)
10. Hernán, M.A., Robins, J.M.: Causal inference. CRC Boca Raton, FL (2010)
11. Kerkhove, S.S., Alechina, N., Dastani, M.: Causes and strategies in multiagent systems. arXiv preprint arXiv:2502.13701 (2025)
12. Lattimore, F., Lattimore, T., Reid, M.D.: Causal bandits: Learning good interventions via causal inference. Advances in neural information processing systems **29** (2016)
13. Miao, W., Geng, Z., Tchetgen Tchetgen, E.J.: Identifying causal effects with proxy variables of an unmeasured confounder. Biometrika **105**(4), 987–993 (2018)
14. Pearl, J.: Causality. Cambridge university press (2009)
15. Rao, A.S., Georgeff, M.P., et al.: Bdi agents: from theory to practice. In: Icmas. vol. 95, pp. 312–319 (1995)
16. Rosenbaum, P.R.: Observational Studies. Springer, New York, 2nd edn. (2002)
17. Sharma, A., Kiciman, E.: Dowhy: An end-to-end library for causal inference. arXiv preprint arXiv:2011.04216 (2020)
18. Simoes, F.N.F.Q., Dastani, M., Ommen, T.v.: Fundamental properties of causal entropy and information gain. In: Locatello, F., Didelez, V. (eds.) Proceedings of the Third Conference on Causal Learning and Reasoning. Proceedings of Machine Learning Research, vol. 236, pp. 188–208. PMLR (01–03 Apr 2024), https://proceedings.mlr.press/v236/simoes24a.html
19. Simoes, F.N., Feigenbaum, I., Dastani, M., van Ommen, T.: The minimal search space for conditional causal bandits. arXiv preprint arXiv:2502.06577 (2025)
20. Simoes, F.N.F.Q., Dastani, M., van Ommen, T.: Causal entropy and information gain for measuring causal control. In: European Conference on Artificial Intelligence. pp. 216–231. Springer (2023)
21. Spirtes, P., Glymour, C.N., Scheines, R.: Causation, prediction, and search. MIT press (2000)
22. Thompson, W.R.: On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. Biometrika **25**(3/4), 285–294 (1933)
23. Zheng, Y., Huang, B., Chen, W., Ramsey, J., Gong, M., Cai, R., Shimizu, S., Spirtes, P., Zhang, K.: Causal-learn: Causal discovery in python. Journal of Machine Learning Research **25**(60), 1–8 (2024)