Proceedings of the Twenty-First AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE 2025)

Accounting for Incomplete and Partially-ordered Observations within Classical Model-based Plan Recognition for Sensemaking of Information Storygames

Jennifer M. Nelson[1*], **Mica Gardone**[1], **Pablo Sauma-Chacón**[1,2], **Rogelio E. Cardona-Rivera**[2]

Laboratory for Quantitative Experience Design
[1]Kahlert School of Computing, University of Utah, 50 Central Campus Dr., Salt Lake City, UT 84112 USA
[2]Division of Games, University of Utah, 332 S 1400 E, Bldg. 72, Salt Lake City, UT 84112 USA
{jennifer.m.nelson | m.gardone | pablo.saumachacon | r.cardona.rivera}@utah.edu

Abstract

We formalize the sensemaking needed to play *information storygames* as a task grounded in automated plan recognition, and outline extensions to a baseline model thereof needed to account for a key part of the information storygame-play loop: non-linear discovery of ambiguous information. This novel problem setting is non-trivial—mechanically simulating this narrative sensemaking requires piecing together incompletely-specified events to reason about the means through which particular ends were achieved in the virtual world. Our work readies plan recognition systems for the task by extending a foundational compilation of *plan recognition as planning* to cover partially-ordered and lifted observations of both actions and facts. While state-of-the-art plan recognizers approximate piecemeal aspects of these features, they do so in isolation of each other and by appealing to disparate and complex frameworks. In contrast, we achieve these functions within a classical planning-based framework. Our results confirm that, while slower, our approach never has more (and often has fewer) false positives than the baseline model in predicting the ground truth plan being executed. We discuss our findings in the context of future work toward better simulating human information storygame sensemaking.

Code — https://github.com/qed-lab/Plan-Recognition-for-Information-Storygames

Datasets — https://osf.io/s3rwz/

Introduction

This paper presents an advance in classical model-based plan recognition applied to reason over partially-ordered and incomplete observations of states and events that have transpired within a virtual world. This kind of reasoning is key to our development of artificial intelligence (AI) systems capable of simulating the human sensemaking involved in playing *information games*, where the goal is to understand a complex artifact by acquiring information, deducing entailed information, and using what has been inferred to guide further acquisition of new information (Francis 2019). We target information *storygames*, where the artifact to understand is a past sequence of events on the basis of deliberately arranged game elements (e.g., plot beats, characters, props, locations) that "suggest a story" (Stewart 2015).

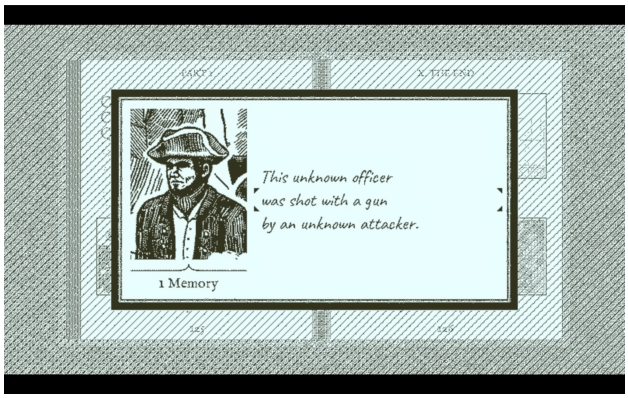

Figure 1: A screenshot from Obra Dinn, depicting the primary player challenge: identify who died, by whom's hand, and by which means, for every member of the titular ship's sixty-person crew. Our goal is to build AI systems that simulate the requisite mental reasoning players engage in.

Simulating the sensemaking that players engage in to play information storygames is non-trivial. Consider our motivating example: the information storygame *Return of the Obra Dinn* (hereafter, "Obra Dinn"; Pope 2018). In it, the merchant ship *Obra Dinn*, which departed from England five years before the storygame setting's time, has mysteriously drifted into port with damaged sails and no visible crew. As the player, you are tasked with determining the fate of the original 52 crewmen and 8 passengers onboard; whether it be their current whereabouts or the combined cause, manner, and perpetrator of their death (Figure 1).

We formalized this task as a combined problem of *plan, activity, and intent* recognition (PAIR, Sukthankar et al. 2014), relying on computer vision to extract structured information from visual scenes (e.g., Johnson et al. 2015; Herzig et al. 2018) and symbolic planning-based recognition to find sequences of actions that bridge them (e.g., Cardona-Rivera and Li 2016; Martens, Cardona-Rivera, and Cohn 2020). However, we quickly discovered that current plan recognition architectures are not suited to processing the kinds of observations players must make in order to mentally re-

[*]Now with Microsoft Corporation.

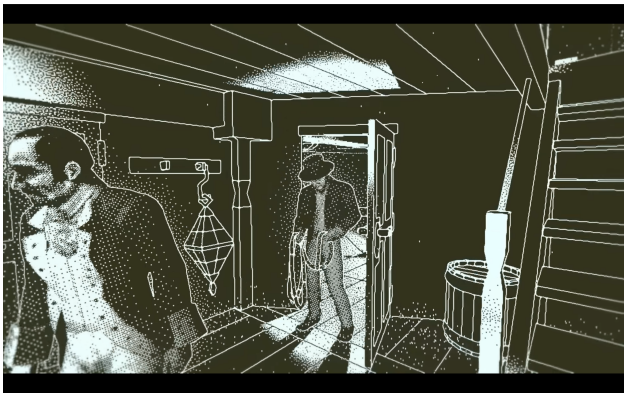

Figure 2: A screenshot from Obra Dinn, depicting a *memory scene* that is triggered by the player when inspecting a victim on the *Obra Dinn*. The memory corresponds to the creature's death, and is preceded by an audio-only rendition of the event(s) immediately prior to displayed situation.

construct the events that transpired during the *Obra Dinn*'s journey. State of the art model-based plan recognition systems predominantly rely on *compilation*, as originally defined by Ramírez and Geffner (2009). In this approach, the recognition problem is solved by compiling (i.e., systematically remodeling) observations into special-purpose goals that a classical planner can work toward to find (recognized) plans that *satisfy* the observations. The problem is that such approaches are unable to adequately process *non-linearly* obtained observations of *ambiguous* information, which are central to information storygame play (Cook 2020).

We therefore extended the baseline plan recognizer by Ramírez and Geffner (2009) to admit *partially-ordered* and *incomplete* observations of both actions and fluents (i.e., state literals), in order to support the rich simulation of a player's mental reconstruction of complex event sequences.

Contributions The baseline Ramírez and Geffner model can only solve recognition problems involving totally-ordered observations of fully-ground actions. Our work generalizes their original notion of what an observation *is* from actions to fluents and from single elements to groups of them, and (re)defines what it means to satisfy observations in this context. And while other scholars have extended the (2009) baseline model to handle some of these facets, no work targets the full extent of our generalization while parsimoniously remaining within classical models.

Like the baseline model, we present a compilation of our expanded representation (i.e., the new observation types) to a classical planning problem in algorithmic detail. Further, we informally describe how our compilation produces classical planning problems that, when solved, constitute a solution to our expanded recognition problem. Finally, we present empirical evidence from a benchmarking study that demonstrates how our method – in addition to never obtaining more false positives than Ramírez and Geffner – often obtains *fewer* false positives, at the cost of a moderate increase in runtime speed.

While there remains more to be done to increase the robustness of our approach, these findings are a modest but critical step toward the ambitious goal of modeling narrative sensemaking in the context of information storygames.

Related Work

We seek to solve the problem of simulating sensemaking that information storygame players engage in; that is, computationally modeling a person's mental re-construction of a storygame's underlying *event model* (cf. Radvansky and Zacks 2017). The most relevant related work concerns alternative model-based plan recognition techniques that one might employ for the sensemaking model we seek. To anticipate: no existing work aims for the same end modeling goal in the same manner of algorithmic technique, although several get close enough to distinguish here.

The task of recognizing a plan performed by a rational agent from its observations has been tackled through the use of off-the-shelf planners as an extension of classical planning (Ramírez and Geffner 2009). This approach has been further extended through the use of probabilistic models to instances in which noisy observations might be introduced into the system due to suboptimal pathing (Ramírez and Geffner 2010). Aineto, Jimenez, and Onaindia (2020) tackled a similar problem in which sensors of a model might provide noisy observations through the use of a probabilistic sensor model and joint likelihood of the obtained observations. Vered, Kaminka, and Biham (2016) described a system for online goal mirroring, which continuously updates its goal recognition by updating the observations taken by the system and uses a planner to predict the plan that would be followed up by observed agent. The obtained plans are compared through a ratio-based heuristic, rather than difference-based, which professedly matches better than behavior of human movement which tends towards optimal pathing in continuous space. Sohrabi, Riabov, and Udrea (2016) further extended the allowable observations through a Bayesian framework to include fluents, even in the presence of unreliable observations. Within epistemic domains, Shvo et al. (2020) described an application extending plan recognition techniques to epistemic domains with multiple agents, by transforming these problems into planner-independent epistemic planning problems.

Overall, these approaches tend to focus on two kinds of "unreliability": (1) observations that are *noisy* (an observation is "noisy" if it appears in the observation sequence and *no plan* for the goal should generate that observation); and (2) observations that are *missing* (an observation is "missing" if it does *not* appear in the observation sequence and the plan for the goal should generate that observation). Our work also focuses on these kinds of unreliability. However, unlike them, we stay entirely within the original classical framework developed by Ramírez and Geffner (2009) while *also* permitting a new kind of unreliability: *incompletely*-sensed information, modeled as observations with lifted parameters. We note that Sohrabi, Riabov, and Udrea could ostensibly do so too, since fluent observations (which they *do* handle) are arguably general enough to accommodate this.

However, they do not present a procedure to solve this directly, whereas we do. Finally, we permit partially-ordered sets of observations, whereas Sohrabi, Riabov, and Udrea (2016) restrict their attention to totally-ordered observations.

Various approaches have sought to solve PAIR problems in the context of storygames. Cardona-Rivera and Young (2015) evaluated the use of symbolic plan recognition (as planning) systems within virtual environments, in contrast to other existing statistical-based approaches (e.g., Min et al. 2017). Similarly, Farrell and Ware (2020) proposed a technique for inferring the goals and beliefs of agents within stories. Meanwhile, Siler and Ware (2023) described an approach to handle *knowledge* goal recognition within interactive narratives—i.e., the goals of an agent that are related to acquiring information about its environment; by representing the common ground known to an agent and the system, the agent's actions can be evaluated with respect to their knowledge to discern potential knowledge goals they might have. These examples showcase the relevance of observations – and plan recognition as a whole – as a driving force within epistemic interactive narrative domains. However, the main difference between those and our work stems from our emphasis in broadening the applicability of plan recognition by permitting additional kinds of observations to be processed. Further, instead of building a story or trying to understand the mental model of actors in a scene, our problem is centered on understanding the scene itself.

Background

Information storygames are effective to the degree a player can acquire the knowledge needed to understand a ground truth sequence of events through an iterative cycle of exploration and inferencing (Cook 2020). In effect, these games invite players to discover a ground truth narrative through clues. To model this ground truth, we adopt the paradigm of *narratives as plans* (Cardona-Rivera et al. 2024).

This paradigm operationalizes "narrative sensemaking" – the task of constructing a consistent mental and event model of a story on the basis of tacitly-available and overtly narrated information – in terms of plans and their construction processes (Cardona-Rivera and Young 2019). Thus, agents within this paradigm should be able to use plan-based reasoning to model not just the events of the story, but also unstated interactions and events that are entailed by what they perceive of the narrative world. Agents must therefore be able to make rational inferences about the narrative and its world with information they have access to (or are given).

Formalisms: Plan and Action Recognition

Our project formalizes sensemaking of information storygames as a combined *Plan* and *Action Recognition* challenge (Sukthankar et al. 2014).

In plan recognition, an *observer* agent aims to predict the future behavior of an *actor* agent given a sequence of observations of the actor's past behavior. Behavior is modeled as a sequence of actions, or *plans*, which transform an agent's initial state of the world to some agent-intended goal state. Plan recognition assumes there is a set of possible goals an actor cares to achieve and predicts: (a) the goals that explain an actor's actions, and (b) the actions the actor will subsequently effect to achieve them.

Listing 1: Example `move` operator in the Planning Domain Definition Language, requiring room adjacency to execute.

```
(:action move ;; via adj(acent) loc(ations)
 :parameters (?a - agent ?from - loc ?to - loc)
 :precondition (and (at ?a ?from) (adj ?from ?to))
 :effect  (and (not (at ?a ?from)) (at ?a ?to)))
```

Whereas plan recognition focuses on recognizing the relationship *between* units of behavior, *activity* recognition focuses on recognizing a *single* unit of behavior on the basis of lower level sensor readings. We focus on *action recognition* (Poppe 2010), which considers the task of discretizing sensor readings of human behavior into coherent units of action that themselves could be input to a plan recognition system.

In this section, we detail the prior work we build upon within plan recognition. To anticipate, this paper focuses on a novel extension to plan recognition systems that are based on a *domain theory*, which relies on a classical planning-based domain model of the actor to transform the observed actions into goals that the observer (an automated planning agent) must satisfy (Ramírez and Geffner 2009). Our novel extension is directly motivated by our intended use of *scene graph*-based action recognition systems (Herzig et al. 2018).

We review the domain-based approach to plan recognition below, briefly covering the classical planning formalism it depends on. We then give an overview of the scene graph representation that motivates our contributions herein.

Classical Planning We rely on the formulation of plan recognition as classical planning, itself a model of problem solving wherein agent actions are fully observable and deterministic. Classical problems are typically represented in a STRIPS-like formalism (Fikes and Nilsson 1971). A planning problem is a tuple $P = \langle F, I, A, G, \mathsf{f}_{\text{cost}} \rangle$, where F is a set of fluents, $I \subseteq F$ is a closed world initial state, $G \subseteq F$ is the set of goal conditions, and A is a set of actions.

An action a is a tuple $\langle \text{TYP}(a), \text{PRE}(a), \text{ADD}(a), \text{DEL}(a) \rangle$, representing the act type label and the precondition, add, and delete lists respectively, which are subsets of F. An action a is applicable in a state s if $\text{PRE}(a) \subseteq s$; applying said action in the state results in a new state $s' = (s \setminus \text{DEL}(a)) \cup \text{ADD}(a)$ and incurs a cost per the function $\mathsf{f}_{\text{cost}} \colon A \to \mathbb{R}^+$. Actions are instances of template *operators* that are identified by act type, with parameters that feature across its precondition and effect (i.e., templetized add/delete lists); Listing 1 illustrates an example `move` operator, written in the Planning Domain Definition Language (PDDL, McDermott et al. 1998).

The solution to a planning problem P is a plan $\pi = [a_1, ..., a_m]$; i.e., a sequence of actions $a_i \in A$ that transforms the problem's initial state I to a state s_m that satisfies the goal—$G \subseteq s_m$. The cost of a plan $c(\pi)$ is $\Sigma \mathsf{f}_{\text{cost}}(a_i)$ for $1 \leq i \leq m$. An *optimal* plan π^* is a plan that solves P and minimizes the overall cost:

$$c(\pi^*) = \min_{\pi \in \text{SOLVE}(P)} c(\pi)$$

A plan segment is a subsequence of a plan, denoted $\pi|_j^k = [a_j, ..., a_k]$ for $a_i \in A$ $(1 \leq j \leq k \leq m)$. If $j = k$, the subsequence is the unary selected element; i.e., $\pi|_j^j = [a_j]$.

The execution $\text{TRACE}(\pi, I) = [I, a_1, s_1, ..., a_m, s_m]$ of plan π from initial state I is defined as the alternating sequence of states and actions, starting with I, such that s_i results from applying $a_i \in \pi$ to state s_{i-1}. For brevity, we denote the *planning domain* model as $D = \langle F, I, A, \mathsf{f}_{\text{cost}} \rangle$, which can be instantiated into a planning problem by adding goal conditions: $D[G] = P = \langle F, I, A, G, \mathsf{f}_{\text{cost}} \rangle$.

Classical Model-based Plan Recognition The original formulation of the domain model-based recognition task extended the definition of a classical planning domain D with two elements. The first element is a set of possible goals $\mathcal{G} = \{G_1, \ldots, G_\ell\}$ that (we assume) the actor being observed cares to achieve. Thus, a plan recognition problem encodes a *set* of planning problems; namely, the set obtained by instantiating $D[G]$ for every $G \in \mathcal{G}$. The second element is a sequence of observations $O = [o_1, \ldots, o_n]$ that the observer must use to deduce the actor's intended goal and subsequent actions toward it. In the classical definition, each $o_i \in A$ is the ith observation of the actor's plan π, itself a fully-ground action that has been executed.

A solution to this problem can be obtained by assuming the actor is behaving optimally with respect to plan cost, and computing a set of *candidate recognized plans* $\hat{\Pi}$: optimal plans that *satisfy* the observations at no extra cost. Satisfaction is defined as follows. Let π be a plan with action indices $i = [1, ..., m]$ and O be an observation sequence with indices $j = [1, ..., n]$. The plan π satisfies the sequence O iff there exists a *monotonically increasing* function f that maps observation indices j onto action indices i such that: $o_j = a_{(i=f(j))}$. The function guarantees that the order of observations is preserved in the plan that satisfies them.

To compute the candidate plans $\hat{\Pi}$ via planning, we transform observations within a classical plan recognition problem $R = \langle F, I, A, \mathsf{f}_{\text{cost}}, \mathcal{G}, O \rangle$ into special-purpose goals G^+ to satisfy via special-purpose actions A^+, which depend on corresponding extra fluents F^+. That is, $\forall o_i \in O$, we:

1. Create an *observation fluent* p_{o_i} and add it to F^+ and G^+
2. Create an *observation action* a_{o_i}, defined as...

$$\begin{aligned} \text{TYP}(a_{o_i}) &= \text{TYP}(o_i) \\ \text{PRE}(a_{o_i}) &= \begin{cases} \text{PRE}(o_i) & \text{if } i = 1 \\ \text{PRE}(o_i) \cup p_{o_{i-1}} & \text{otherwise} \end{cases} \\ \text{ADD}(a_{o_i}) &= \text{ADD}(o_i) \cup p_{o_i} \\ \text{DEL}(a_{o_i}) &= \text{DEL}(o_i) \end{aligned}$$

...and add it to A^+ with $\mathsf{f}_{\text{cost}}(a_{o_i}) = \mathsf{f}_{\text{cost}}(o_i)$.

Importantly, the actions in A^+ *replace* the original actions of the same type in A. For clarity, we denote the new set of actions as $\mathcal{A} = A^- \cup A^+$, where A^- is the set of actions that did not feature in any observation; i.e., $A^- = \{A \setminus O\}$.

Further, the special-purpose goals are *appended* to every assumed possible goal. This resulting new set of assumed goals $\mathcal{G}'$ can be described in set-builder notation as:

$$\mathcal{G}' = \{G' \mid \forall G \in \mathcal{G} \colon G \cup G^+\}$$

Subsequently, we use the original recognition problem to construct a new one, of the form:

$$R' = \langle F \cup F^+, I, \mathcal{A}, \mathsf{f}_{\text{cost}}, \mathcal{G}', \varnothing \rangle$$

Here, each observation has been "compiled away" into a special-purpose action that contains an additional effect (i.e., p_{o_i}), whose sole purpose is to reify the fact that the action was observed. Further, if an observation is preceded by another, the earlier action's effect becomes a precondition of the later one; this guarantees observation actions appear in the order their corresponding observations were registered.

The resulting new problem R' captures the semantics of the original, while also being structurally equivalent to a set of planning problems; namely (and as before), the set obtained by instantiating the new domain $D' = \langle F \cup F^+, I, \mathcal{A}, \mathsf{f}_{\text{cost}} \rangle$ with every new goal $G' \in \mathcal{G}'$.

Thus, after compilation there are two sets of assumed goals: $\mathcal{G} = \{G_1, \ldots, G_\ell\}$ and $\mathcal{G}' = \{G'_1, \ldots, G'_\ell\}$. Each set supports instantiating ℓ individual planning problems over D and D', respectively. The observer agent can then predict the recognized plan(s) by solving 2ℓ problems; i.e., solving the *baseline problem* $D[G]$ to yield plan π^* and its counterpart *observation-compiled problem* $D'[G']$ to yield plan π', for each of the ℓ goals in $\mathcal{G}$ and $\mathcal{G}'$, respectively. A plan π^* is deemed *recognized* iff it can account for the observations at no extra cost; i.e., iff $c(\pi^*) = c(\pi')$.

This formulation of domain-based plan recognition reflects several implicit assumptions about the observed actor (Masters and Sardina 2019); namely, that they are (a) behaving optimally with respect to plan cost (as mentioned), (b) either unaware they are being observed – i.e., this is a *keyhole* recognition setting (Carberry 1990) – and/or behaving honestly, and (c) have a single goal in mind.

Scene Graph-based Action Recognition Our work is motivated by the use of scene graph-based action recognition systems. Specifically, we expect to adopt the approach by Herzig et al. (2018) to map images onto *scene graphs* (Krishna et al. 2016). A scene graph is a directed graph consisting of *objects*, *attributes* for objects, and *relationships* between objects. Scene graphs are semantically grounded into images via bounding boxes, as depicted in Figure 3.

We envision converting scene graphs into logical fluents that capture the depicted attributes and relations, as proposed by Cardona-Rivera and Li (2016). As they note, it is possible this parsing is noisy, yielding fluents that are not interoperable with the domain definitions used during planning. For now, we expect to make a similar simplifying assumption that if a literal is said to exist from the parser, but is not in the domain, it is discarded. If all image literals are discarded, the photo is removed.

Problem Setting: Return of the Obra Dinn

We discuss next what the narrative sensemaking problem looks like through the lens of Obra Dinn, as depicted pictorially in Figure 4. We note that the scope of the full sensemaking problem is larger than what we target in this paper, which presents a solution to a component of it. At the same time, the purpose of presenting the full problem is so that we may better contextualize our contribution and progress toward our goal of modeling narrative sensemaking.

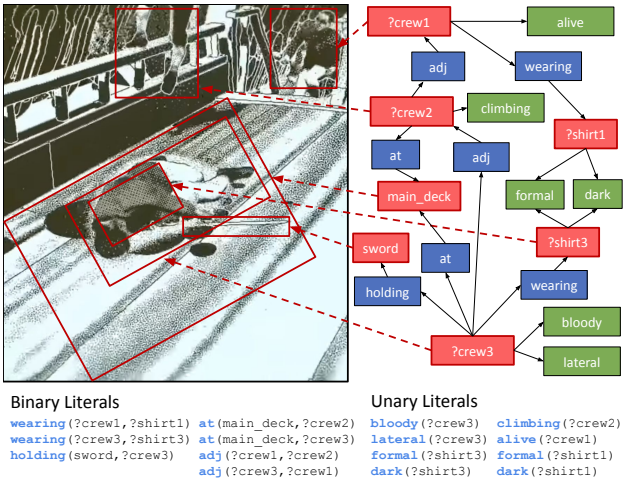

Figure 3: An example scene graph and its bounding box-based grounding, as parsed from an image (top left) with extracted logical literals (bottom). The scene graph contains objects (in red), relations (in blue) and attributes (in green).

As input, a narrative sensemaking system is given an audiovisual stream captured by a virtual camera (e.g., Galvane et al. 2014) within Obra Dinn during gameplay. This stream would eventually be distilled to a set of appropriately staged photographs (cf. Louarn, Christie, and Lamarche 2018) that information-theoretically cover a virtual space of interest. Each image in this set is then compiled to a scene graph grounded through bounding boxes (Herzig et al. 2018), which is then distilled into a set of literals that describe what the player observes (e.g., Cardona-Rivera and Li 2016).

Literals ought to be grounded if there is certainty to what was presented. For example, if a player sees a gun has been fired, they will not necessarily know by whom. Unknown actions and fluents are ambiguous to the degree their requisite parameters are unbound. If possible, orderings over actions and fluents are established, but in an information game this ordering might not be immediately obvious. Ambiguous orderings must then be theorized to entertain the possible topological sorts. For example, if a gun powder barrel was destroyed, did it happen at the same time as a gunshot was heard? or did a gun fire *and then* the barrel exploded? As more images are processed, information may be filled in or revealed. The existing model of events might be revised and processed again to make predictions or backfill inferences. Eventually, we expect the system to converge on the series of actions that represents the true fate of the *Obra Dinn*.

Problem Statement We present a solution to the plan recognition problem defined over partially-observed actions, fluents, and orderings in some task domain. In specific, we implement the observation compiler component and its connection to a classical planner as depicted in Figure 4.

Method

To accommodate incomplete and partially-ordered observations, we relax one assumption by Ramírez and Geffner (2009). This relaxation allows us to include new kinds of observations into the plan recognition problem and produce a compilation that translates to the desired planning problems.

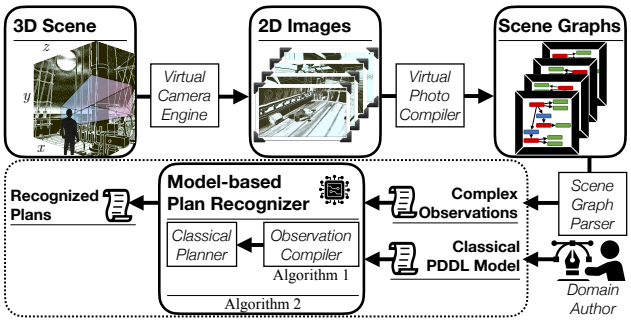

Figure 4: A pictorial overview of the sensemaking narrative problem. Our contribution is the Model-based Plan Recognizer (i.e., Algorithms 1 and 2 within the dotted line).

Incomplete and Partially-ordered Observations

The relaxed assumption concerns that the observations $O = [o_1, \ldots, o_n]$ need to be totally ordered and grounded actions. Instead, we allow observations to be either an observed action or a set of observed fluents. We also allow observations to be partially-ordered with respect to others in the sequence. Further, we allow both action and fluent observations to be incompletely specified (i.e., lifted); e.g., in our method, it is possible to observe both `(move ?c GunDeck OrlopDeck)` and `(at ?c MainDeck)`. We assume action observations have disambiguated the act type; otherwise, the situation requires *action* recognition, which is of interest but beyond our scope.

We relax this assumption by defining **observation groups**, which impose constraints on the observations they contain. In what follows, we describe two unit observations that are the basis of our approach, the three types of observation groups we can form, and what it means for a plan to **satisfy** an observation and groups thereof.

An observation group is a (possibly unary) collection of other observations. Because groups can nest within each other, we use Θ to denote the outer, *containing* group, and θ to denote any inner, *member* observation(s); any such member could be a group. Further, observation satisfaction of an outer group is defined in terms of observation satisfaction of its nested member(s) by a plan *through a plan segment*.

Simple Observations The basis unit of our method (the unit over which observation groups may be formed) is a single, *simple observation*. We support two kinds of simple observations: *action observations* and *fluent observations*. Thus, we begin by describing the satisfaction of simple observations in terms of a plan through a segment.

Definition 1 (Action observation satisfaction). An action observation o paired with action $a \in A$ is *satisfied by* a plan π through segment $\pi|_j^k$ iff $a = a_i$ for some $a_i \in \pi|_j^k$.

A fluent observation corresponds to a (possibly unary) *set* of fluents, and is satisfied by plan segments that cover a "time period" where those fluents are true for some state.

Definition 2 (Fluent observation satisfaction). A fluent observation o paired with a set of fluents $F_o \subseteq F$ is *satisfied by* a plan π starting with initial state I through segment $\pi|_j^k$ iff $F_o \subseteq s_i$ for some s_i $(j \leq i \leq k) \in \text{TRACE}(\pi, I)$.

Importantly, the actions of the plan segment we reference in Def. 2 need not actually contribute to the observed fluents for this notion of satisfaction. The plan segment merely marks the portions of a plan's execution within which the respective fluents were observed. It may be that F_o was true in the initial state, but not observable until much later. Our intent here is to rule out candidate recognized plans whose traces never co-occur with the fluents in F_o being true.

Ordered, Unordered, and Option Groups The observation groups impose constraints on its contained members. We define three such groups below.

An *ordered group* can only be satisfied by a plan segment if that segment can be split into sub-segments that satisfy the contained members *in order*.

Definition 3 (Ordered group satisfaction). An ordered observation group $\Theta_< = [\theta_1, \ldots, \theta_n]$ is a totally ordered (possibly unary) sequence of observation groups and/or simple observations. A plan π satisfies $\Theta_<$ through segment $\pi|_j^k$ iff there exists a *monotonically non-decreasing* function $f\colon [1, n+1] \mapsto [j, k+1]$, with $f(n+1) = k+1$, which maps members of $\Theta_<$ to (sub-)segments of $\pi|_j^k$ such that:

$$\begin{aligned} \text{plan segment} \quad & \pi\Big|_{f(i)}^{(f(i+1)-1)} = \big[a_{f(i)}, \ldots, a_{(f(i+1)-1)}\big] \\ \text{satisfies} \quad & \theta_i \in \Theta_<. \end{aligned}$$

Akin to its role within the baseline model, the function f ensures the ordering of observations is preserved in the recognized plan. However, unlike the baseline model, said function is monotonically non-decreasing,[1] which is needed to handle embedded members whose internal elements are unordered. The function f maps subsequent group members to correspondingly subsequent plan segments, meaning that $f(i)$ marks the beginning of θ_i's plan segment. We disallow gaps in plan segments, such that θ_i's segment ends immediately before θ_{i+1}'s segment begins, and θ_n's segment ends where the whole plan segment ends.

In contrast to ordered groups, *unordered groups* impose no ordering constraints on members. Instead, they only impose *satisfaction* constraints; i.e., an unordered group is satisfied when all its members are. When an unordered group is embedded in an ordered one, the unordered group forms the *partiality* of a *partial ordering*.

Definition 4 (Unordered group satisfaction). An unordered observation group $\Theta_\wedge = \{\theta_1, \ldots, \theta_n\}$ is a *set* of observation groups and/or simple observations with no ordering constraints defined among them. A plan π satisfies $\Theta_\wedge$ through the plan segment $\pi|_j^k$ iff the segment satisfies all $\theta_i \in \Theta_\wedge$.

Lastly, *option* groups, which – unlike the others – may contain only *simple* observations, are satisfied if at least one of the contained members is satisfied.

[1] f is monotonically non-decreasing iff $\forall(x{>}y)\colon\ f(x){\geq}f(y)$.

Definition 5 (Option group satisfaction). An option group $\Theta_\oplus = |o_1, \ldots, o_n|$ is a *set* of simple observations where it is uncertain which of them was the true observation. A plan π satisfies $\Theta_\oplus$ through the plan segment $\pi|_j^k$ iff the segment satisfies *at least one* $o_i \in \Theta_\oplus$.

Option groups are how we describe a set *possible* observations, and we use them to model *incomplete* (i.e., lifted) observations: to us, an incomplete observation is equivalent to an option group of all its possible interpretations (i.e., groundings) in the planning domain. Further, plans that satisfy more members are *not* considered more likely solutions than those that satisfy fewer members.

We can now formally state our plan recognition problem. It is largely akin to classical domain-based plan recognition, but replaces the total-order, fully-ground input observations with the more general observation groups as defined so far.

Definition 6 (Plan recognition problem over groups). A plan recognition problem over observation groups is a tuple $\mathfrak{R} = \langle F, I, A, \mathsf{f}_{\text{cost}}, \mathcal{G}, \Theta\rangle$, where $D = \langle F, I, A, \mathsf{f}_{\text{cost}}\rangle$ is the planning domain; $\mathcal{G}$ is the assumed set of possible goals $G_i \subseteq F$ the actor being observed cares to achieve; and Θ is a group of n observations θ_i $(1 \leq i \leq n)$.

Compilation to Planning

Like before, $\mathfrak{R}$ encodes a set of n planning problems. We therefore pursue a similar strategy of compiling $\mathfrak{R}$ into a recognition problem variant $\mathfrak{R}'$ with observation groups compiled away. This (again) yields 2ℓ problems, obtained by instantiating $D[G]\colon \forall G \in \mathcal{G} = \{G_1, \ldots, G_\ell\}$ and their counterparts $D'[G']\colon \forall G' \in \mathcal{G}' = \{G'_1, \ldots, G'_\ell\}$. A *solution* to $\mathfrak{R}$ is a set of candidate recognized plans that satisfy the observations. We must therefore ensure our compilation constructs the recognition problem variant $\mathfrak{R}'$ appropriately.

The compilation ought to guarantee that, for all planning problems $D'[G']$ contained within $\mathfrak{R}'$, their corresponding solution π': (a) satisfies every $\theta_i \in \Theta$, (b) respects Θ's ordering constraints, and (c) ensures that exactly *one* observation is satisfied in every respective option group nested within.

Our compilation procedure is presented in Algorithm 1. It processes the observation group in two general steps, as we did earlier. First (Line 4), we create the additional *explanation fluents*, which reify observations into F^+ and G^+ as additional goals to strive for. Second (Line 13), we define *explanation actions* e_{o_i}, akin to the aforementioned *observation actions* but which can be used to explain both observed actions and fluents. The combined explanation fluents and actions ensure an observation is reified only *after* its predecessors (as before), and that there is only one explanation per simple observation and per option group.

For clarity, we let $\text{NESTED}(\Theta)$ denote the set of all simple observations that are members of group Θ or any of its contained subgroups (Line 3). Further, $\text{OPTIONID}(\Theta_\oplus)$ denotes a unique identifier for the given option group, used to combine the reification of the respectively contained simple observations under one common explanation fluent named $p_{\text{OPTIONID}(\Theta_\oplus)}$ (Line 6). Finally, $\text{PREC}(o_i)$ denotes the set of simple observations NESTED within any group Θ^* that precedes the group Θ that o_i is NESTED within (Lines 16, 23).

Algorithm 1: COMPILE: $\mathfrak{R} \rightarrow \mathfrak{R}$. Transformation of the plan recognition problem over groups to a recognition problem variant with observations compiled away.

Input: $\mathfrak{R}$, a plan recognition problem over observation groups, composed of: a planning domain $D = \langle F, I, A, \mathsf{f}_{\text{cost}} \rangle$; a set of ℓ assumed actor goals $\mathcal{G}$; and a group of n observations Θ.
Output: $\mathfrak{R}'$, a plan recognition problem with: $\Theta = \varnothing$; a planning domain $D' \supset D$; and a set of assumed actor goals $\mathcal{G}' \supset \mathcal{G}$.

1: Let F^+, G^+, and $\mathcal{G}' \leftarrow \varnothing$ {Compilation constructs.}
2: Let $\mathcal{A} \leftarrow A$ {Define compiled actions from baseline.}
3: **for all** $o_i \in \text{NESTED}(\Theta)$ **do** {All simple observations}
4: Create *explanation fluent* p_e
5: **if** o_i is a member of an option group $\Theta_\oplus$ **then**
6: Let $p_e \leftarrow$ shared option fluent $p_{\text{OPTIONID}(\Theta_\oplus)}$
7: **else**
8: Let $p_e \leftarrow$ unique fluent p_{o_i}
9: **end if**
10: $F^+ \leftarrow \left(F^+ \cup p_e\right)$ {Add it as a special-purpose fluent.}
11: $I \leftarrow (I \cup \neg p_e)$ {Observations begin unexplained.}
12: $G^+ \leftarrow \left(G^+ \cup p_e\right)$ {Add it as a special-purpose goal.}
13: Create (an empty) *explanation action* $e_{o_i} = \langle \varnothing, \varnothing, \varnothing, \varnothing \rangle$
14: **if** o_i is a *fluent observation* corresponding to $F_{o_i} \subseteq F$ **then**
15: $\text{TYP}(e_{o_i}) = \texttt{obs-fluent-}F_{o_i}$
16: $\text{PRE}(e_{o_i}) = F_{o_i} \cup \{\neg p_e\} \cup \left\{p_{o_{\text{prec}}} | o_{\text{prec}} \in \text{PREC}(o_i)\right\}$
17: $\text{ADD}(e_{o_i}) = \{p_e\}$
18: $\text{DEL}(e_{o_i}) = \varnothing$
19: Map $\mathsf{f}_{\text{cost}}(e_{o_i}) = 0$ {Explaining fluents costs nothing.}
20: $\mathcal{A} \leftarrow \mathcal{A} \cup e_{o_i}$ {Add explanation action.}
21: **else** {o_i is an *action observation* corresponding to $a \in A$}
22: $\text{TYP}(e_{o_i}) = \text{TYP}(a)$
23: $\text{PRE}(e_{o_i}) = \text{PRE}(a) \cup \{\neg p_e\} \cup \left\{p_{o_{\text{prec}}} | o_{\text{prec}} \in \text{PREC}(o_i)\right\}$
24: $\text{ADD}(e_{o_i}) = \text{ADD}(a) \cup \{p_e\}$
25: $\text{DEL}(e_{o_i}) = \text{DEL}(a)$
26: Map $\mathsf{f}_{\text{cost}}(e_{o_i}) = \mathsf{f}_{\text{cost}}(a)$
27: $\mathcal{A} \leftarrow ((\mathcal{A} \setminus \{a\}) \cup e_{o_i})$ {Swap a for explanation action.}
28: **end if**
29: **end for**
30: **for all** $G \in \mathcal{G}$ **do** {Append special-purpose goals.}
31: $\mathcal{G}' \leftarrow \mathcal{G}' \cup \left\{G \cup G^+\right\}$
32: **end for**
33: **return** $\mathfrak{R}' = \left\langle F \cup F^+, I, \mathcal{A}, \mathsf{f}_{\text{cost}}, \mathcal{G}', \varnothing, B \right\rangle$

Algorithm 1 boasts several features worth discussing. First, the manner in which we create explanation actions for fluent observations (Line 14) is inspired by Sohrabi, Riabov, and Udrea (2016), except that we permit mapping the observation o_i to *multiple* fluents $F_{o_i} \subseteq F$ instead of a single one as they do. Second, we note that mapping an action cost to zero (Line 19) requires either using a (non-classical) *metric* planner (e.g., METRIC-FF as developed by Hoffmann 2003) *or* tracking the costs of these actions and subtracting them from the cost of the observation-compiled solution plan (i.e., $c(\pi')$). Third, all explanation actions have the precondition $\neg p_e$, but add p_e as an effect; as no action *removes* p_e, this means that an observation cannot be explained twice.

As we empirically demonstrate next, our compilation produces a set of planning problems that can be used to solve the plan recognition problem when embedded within the *plan recognition as planning* strategy (Algorithm 2).

Algorithm 2: PLANREC: $\mathfrak{R} \rightarrow \hat{\Pi}$. Computation of the candidate recognized plans that satisfy the input observations.

Input: $\mathfrak{R}$, a plan recognition problem over observation groups, composed of: a planning domain $D = \langle F, I, A, \mathsf{f}_{\text{cost}} \rangle$; a set of ℓ assumed actor goals $\mathcal{G}$; and a group of n observations Θ.
Output: $\hat{\Pi}$, a (possibly empty) set of plans deemed to be recognized on the basis of input observations.

1: Let $\hat{\Pi} \leftarrow \varnothing$ {Candidate recognized plan set.}
2: Let $\mathfrak{R}' \leftarrow \text{COMPILE}(\mathfrak{R})$ {Per Algorithm 1.}
3: **for** $i = 0$ to $n = \|\mathcal{G}\|$ **do** {Plan for each pair of goals.}
4: Baseline $\pi_i \leftarrow \text{SOLVE}(D'[G_i])$, $G_i \in \mathcal{G} \in \mathfrak{R}$
5: Compiled $\pi'_i \leftarrow \text{SOLVE}(D'[G'_i])$, $G'_i \in \mathcal{G}' \in \mathfrak{R}'$
6: **if** $\pi_i, \pi'_i \neq \varnothing \wedge (c(\pi_i) - c(\pi'_i) = 0)$ **then**
7: $\hat{\Pi} \leftarrow \hat{\Pi} \cup \{\pi'_i\}$
8: **end if**
9: **end for**
10: **return** $\hat{\Pi}$

Evaluation

We provide empirical evidence that our compilation transforms a plan recognition problem into a variant that codifies a set of planning problems, which – when solved – allow us to precisely identify the solutions to the *plan recognition problem over observation groups*. Space limitations preclude us from presenting the relevant formal proofs.

Experiment Setup

In our analyses, recognized plans are uniquely identified by the goal they accomplish; i.e., given a goal, the recognized plan will always be the optimal one that satisfies the observations or none at all (Ramírez and Geffner 2009). As such, our experiment is couched in terms of the size of resultant optimal goal sets $\mathcal{G}^*$; i.e., the set of goals respectively accomplished by optimal plans that satisfy the observations. Here, smaller goal sets are more precise than larger ones, and therefore preferred in general.

Hypotheses We hypothesized that the size of the optimal goal set $\mathcal{G}^*_{\text{ours}}$ produced via plan recognition with our compilation will often be smaller, and never larger, than the size of the goal set produced via plan recognition using the baseline Ramírez and Geffner compilation $\mathcal{G}^*_{\text{base}}$, as defined below. We also hypothesized it would take longer to compute our optimal goal set in order to accommodate increased complexity of observation types.

Apparatus We ran all software under a Windows Subsystem for Linux 2 instance with 31 GB of RAM allocated to it. The native hardware used was 32 GB of RAM and an Intel 9700K CPU at 3.6 GHz running Windows 10.

Procedure While conducting an evaluation on the actual Obra Dinn storygame is aspirational for us, we avoided hand-crafting a bespoke evaluation domain to avoid issues with credit assignment, out of concern that any exhibited performance might be in part due to the way we craft such domain (as opposed to the algorithm, which is what we intend to analyze). Instead, to empirically validate

our approach, we benchmarked its performance against fifteen domains compiled by Pereira and Meneguzzi (2017); to the best of our knowledge, these domains are a relatively recently-established benchmarking suite for evaluating compilation-based approaches to plan recognition.

For every plan recognition problem, consisting of a domain D and possible goals $\mathcal{G}$ of which G_{true} is correct, we found an optimal plan for $D[G_{\text{true}}]$, and synthetically generated complex observations from both the plan (**A**) and plan's trace (**A+F**). For each pair, we generated an observation set varying the percentage of observations of unknown order (**U%**) and percentage of observations that were ambiguated by missing a parameter (**B%**). To generate observations, we randomly removed half the optimal plan/trace, and removed 90% of fluents from states in each trace. Lastly, we ambiguated **B%** of action observations by removing one parameter and replacing the observation with an option group of all matching observations; e.g., the action `(eat H)` becomes the option group $\Theta_{\oplus} = |$`(eat A)`$, \ldots,$`(eat Z)`$|$. Ambiguation (**B%**) was only applied to action observations with at least one parameter. We varied **U%** and **B%** over four settings:

1. No Obscuration: **U=0% B=0%**
2. Vary Ambiguation: **U=0% B=25%**
3. Vary Unorderedness: **U=50% B=0%**
4. Vary Both: **U=50% B=25%**

For each problem $\langle D, \mathcal{G}, \Theta \rangle$, and each goal $G \in \mathcal{G}$, we compiled two planning problems:

1. $D'_{\text{ours}}[G'_{\text{ours}}]$, generated via our compilation; and
2. $D'_{\text{base}}[G'_{\text{base}}]$, generated via the baseline Ramírez and Geffner compilation with the following "ignore complexity" strategy: fluent observations and option groups are removed, all unordered groups are reduced to a single member, and unary/empty member groups are flattened until just an ordered group of action observations is left.[2]

We then solved each problem using the FASTDOWNWARD planner configured to use A* with landmark-cut heuristic (Helmert 2006), and compared the optimal costs of both to the optimal cost for $D[G]$ (precomputed). If:

$$c^*(D'_{\text{ours}}[G'_{\text{ours}}]) = c^*(D[G]), \quad \text{we added } G \text{ to } \mathcal{G}^*_{\text{ours}},$$
$$c^*(D'_{\text{base}}[G'_{\text{base}}]) = c^*(D[G]), \quad \text{we added } G \text{ to } \mathcal{G}^*_{\text{base}}$$

Results and Discussion

Due to space limitations, we formally present only four of the studied domains: (a) `intrusion-detection`, a close analog to *Obra Dinn*, focusing on discovering clues about a break-in in a house; (b) `kitchen`, a domain similar to the game *Overcooked*, where knowledge about recipes is gate-kept; (c) `miconic`, a domain that supports encoding puzzle games with varying levels of challenge difficulty; and (d) `easy-ipc-grid`, originally used as a benchmarking domain for the baseline Ramírez and Geffner (2009) plan recognition method, but which has no obvious relevance or mapping to games.

[2]We choose this strategy over strategies that try different orderings/option group members because they would take exponentially longer to solve, requiring as many trials as there are combinations of unordered group orders and option group choices.

Analysis

We removed instances where our simplification strategy resulted in an empty observation set. We saw this occur mostly in the [**A+F U:50%B:25%**] tests. Tables 1 and 2 report the number of perfectly solved problems (Opt) and sub-optimally solved problems with room for improvement (Sub-opt), per observation complexity setting, per domain. These tables also report the average number of observations ($|\Theta|$) per method (Ours v. Base), the average size of the solution set ($|\mathcal{G}^*|$) when improvable, and the average time to compute (whether or not improvable). Error rates indicate a 95% confidence interval. For $|\Theta|$, we defined option groups as being size 1.

We conducted an independent t-test, assuming unequal variance, to compare the sizes of solution sets when improvable. All analyses found statistically significant support ($p < 0.0001$) to reject the null hypotheses in favor of our alternates: that our method reduces the size of the optimal goal set and is thereby more precise, at the cost of an increase in runtime.

Discussion of Results

Our method is overall slower than the original formulation, regardless of solving improvements. We hypothesize that this is due to a larger search space. Using more observations means including more actions in the planning domain, which usually takes longer to compute.

We can also see, where **B:25%**, a noticeable increase in the number of potential improvements to the solutions. The increase in the number of solutions needing improvements indicates that the ambiguations have a major effect on the planner's ability to navigate the search space effectively. More so, that ambiguations are a stronger correlation to worse performance than action unordered-ness. Across **A** and **A+F**, we can see that ambiguating causes solutions to need improvements in at least half, with anomalies in `intrustion-detection` and `miconic` (only **A**).

Limitations

The most notable limitation of the work is that we cannot claim that this system has a correspondence to the human process of sensemaking, for at least two reasons. First, we conducted no comparison to human performance. Second, the evolution from prior work focuses on the *technical* components needed to process the complex observation information. We are certainly interested in approximating the cognitive *process* a player would undergo while playing an information game. Our future work will thus look not just at a *functional* model of narrative sensemaking (i.e., in terms of inputs/outputs), but rather seek correspondence in the moment-to-moment processing of non-linearly discovered ambiguous information.

Another limitation is that while our compilation is capable of predicting a plan more precisely than prior work in the presence of these complex observations, the systems cannot run in an incremental fashion. Future work will explore how to *revise* existing predictions rather than simply running the plan recognition process from scratch.

Configuration				Individual Performance		Relative Performance (± values represent a 95% Confidence Interval)							
	Problem Settings			*Number of Problems Solved*		$\mu_{\lvert\Theta\rvert}$ *for OSPs*		$\mu_{\lvert\Theta\rvert}$ *for SSPs*		$\mu_{\lvert\mathcal{G}^*\rvert}$ *for SSPs*		***Runtime OSPs + SSPs*** (s)	
Domain	U%	B%	n	Opt (***OSPs***)	Sub-opt (***SSPs***)	Base	Ours	Base	Ours	Base	Ours	Base	Ours
intrusion	0%	0%	338	338	0	8.59±0.08	''	n/a	n/a	n/a	n/a	**15.38±0.61**	**14.09±0.61**
	0%	25%	338	326	12	6.00±0.05	8.60±0.09	5.75±0.29	8.25±0.55	2.08±0.18	1.75±0.39	**14.73±0.56**	**86.10±6.73**
	50%	0%	338	330	8	5.61±0.08	8.61±0.08	4.75±0.74	7.75±0.74	**2.12±0.30**	**1.12±0.30**	14.67±0.55	14.13±0.53
	50%	25%	338	288	50	4.41±0.10	8.66±0.09	3.56±0.27	8.18±0.23	**2.26±0.17**	**1.08±0.08**	**34.97±40.43**	**87.11±6.92**
kitchen	0%	0%	122	99	23	5.97±0.69	''	3.00±0.00	''	2.00±0.00	''	4.19±0.16	4.14±0.17
	0%	25%	103	78	25	4.12±0.56	5.96±0.78	2.00±0.00	3.00±0.00	2.00±0.00	''	**4.94±0.42**	**15.28±1.96**
	50%	0%	146	115	31	4.13±0.46	5.98±0.64	2.00±0.00	3.00±0.00	**2.00±0.00**	**1.55±0.27**	**4.03±0.16**	**4.31±0.16**
	50%	25%	144	118	26	3.55±0.33	6.44±0.64	1.58±0.20	3.00±0.00	**2.00±0.00**	**1.58±0.31**	**4.14±0.14**	**14.09±1.69**
ipc-grid	0%	0%	565	480	85	6.87±0.14	''	6.66±0.10	''	2.00±0.00	''	**12.22±0.49**	**9.85±0.41**
	0%	25%	565	435	130	4.86±0.11	6.89±0.16	4.65±0.09	6.66±0.09	**2.06±0.07**	**1.75±0.09**	11.07±0.41	10.68±1.17
	50%	0%	565	457	108	4.23±0.10	6.88±0.15	4.00±0.00	6.65±0.09	**2.12±0.13**	**1.75±0.08**	10.89±0.42	10.39±0.51
	50%	25%	565	410	155	3.35±0.10	6.89±0.16	2.93±0.11	6.71±0.09	**2.08±0.08**	**1.58±0.08**	10.39±0.38	10.47±0.53
miconic	0%	0%	655	626	29	8.70±0.05	8.70±0.05	8.90±0.24	8.90±0.24	2.14±0.13	2.14±0.13	**9.44±0.04**	**8.93±0.04**
	0%	25%	655	557	98	6.08±0.02	8.71±0.05	6.01±0.05	8.71±0.12	**2.08±0.06**	**1.64±0.12**	**9.16±0.04**	**9.34±0.06**
	50%	0%	655	518	137	5.73±0.05	8.73±0.05	5.61±0.11	8.61±0.11	**2.18±0.06**	**1.31±0.08**	**9.14±0.04**	**8.93±0.04**
	50%	25%	655	430	225	4.54±0.08	8.75±0.06	4.19±0.10	8.63±0.08	**2.30±0.07**	**1.26±0.06**	**9.04±0.04**	**9.48±0.08**

Table 1: Evaluation results per domain and setting for **A**ction observations. **U%** is percent of observations placed in an unordered set. **B%** is percent of 'ambiguated' observations. We distinguish between samples perfectly solved by the ignore strategy (Opt) and samples with sub-optimal solutions (sub-opt, SSP). $|\Theta|$ is the observation set size for the specified method and sample group. $|\mathcal{G}^*|$ is the size of the solution set over the SSP samples. Bold indicates a t-test significant difference ($\alpha < .05$). Underlines indicate the smaller of two means. Values with error rates are means with a 95% confidence interval.

Configuration				Individual Performance		Relative Performance (± values represent a 95% Confidence Interval)							
	Problem Settings			*Number of Problems Solved*		$\mu_{\lvert\Theta\rvert}$ *for OSPs*		$\mu_{\lvert\Theta\rvert}$ *for SSPs*		$\mu_{\lvert\mathcal{G}^*\rvert}$ *for SSPs*		***Runtime OSPs + SSPs*** (s)	
Domain	U%	B%	n	Opt (***OSPs***)	Sub-opt (***SSPs***)	Base	Ours	Base	Ours	Base	Ours	Base	Ours
intrusion	0%	0%	338	336	2	8.54±0.19	16.57±0.17	7.50±19.06	17.00±0.00	**2.00±0.00**	**0.00±0.00**	**15.42±0.60**	**54.18±9.46**
	0%	25%	333	278	55	4.17±0.17	16.63±0.19	2.56±0.32	16.38±0.34	**2.60±0.26**	**0.58±0.35**	**14.44±0.54**	**61.41±9.39**
	50%	0%	338	336	2	7.00±0.14	16.58±0.17	6.50±6.35	16.50±6.35	**2.00±0.00**	**0.00±0.00**	**15.04±0.58**	**35.38±7.23**
	50%	25%	335	265	70	3.83±0.14	16.71±0.20	2.44±0.24	16.09±0.31	**2.64±0.31**	**0.82±0.22**	**14.41±0.54**	**37.10±6.80**
kitchen	0%	0%	100	78	22	4.99±0.77	9.72±1.45	2.23±0.27	5.95±0.09	**2.00±0.00**	**0.82±0.22**	**4.48±0.17**	**5.91±1.15**
	0%	25%	81	54	27	3.37±0.50	13.91±1.83	1.33±0.19	5.89±0.13	**2.07±0.11**	**1.00±0.27**	**5.82±0.73**	**11.09±1.59**
	50%	0%	121	96	25	4.82±0.56	11.78±1.38	2.52±0.29	5.92±0.11	**2.00±0.00**	**1.04±0.28**	**4.17±0.10**	**6.17±1.11**
	50%	25%	135	87	48	2.91±0.36	13.30±1.46	1.38±0.14	5.96±0.06	**2.06±0.07**	**0.94±0.13**	**4.45±0.23**	**9.64±1.32**
ipc-grid	0%	0%	564	484	800	6.99±0.17	13.14±0.27	6.40±0.31	12.88±0.15	**2.00±0.00**	**0.30±0.16**	**12.31±0.49**	**10.44±0.43**
	0%	25%	546	333	213	3.26±0.15	13.26±0.36	2.68±0.17	13.04±0.15	**2.44±0.19**	**0.29±0.10**	10.18±0.36	10.44±0.44
	50%	0%	564	468	96	5.46±0.14	13.15±0.27	5.00±0.22	12.92±0.19	**2.03±0.06**	**0.27±0.13**	**11.35±0.44**	**10.45±0.43**
	50%	25%	546	330	216	3.06±0.14	13.21±0.36	2.37±0.14	13.16±0.19	**2.61±0.22**	**0.21±0.09**	10.14±0.36	10.49±0.44
miconic	0%	0%	655	614	41	8.84±0.12	16.99±0.08	7.76±0.42	16.83±0.25	**2.05±0.07**	**1.15±0.41**	**9.46±0.04**	**6.63±0.02**
	0%	25%	653	396	257	4.67±0.13	16.95±0.10	3.40±0.17	17.02±0.14	**2.54±0.09**	**0.50±0.13**	**9.02±0.04**	**6.63±0.02**
	50%	0%	655	595	60	7.30±0.10	16.98±0.08	6.50±0.32	16.90±0.33	**2.20±0.11**	**0.68±0.35**	**9.29±0.04**	**6.63±0.02**
	50%	25%	650	376	274	4.33±0.11	16.91±0.11	3.22±0.14	17.07±0.13	**2.55±0.10**	**0.59±0.13**	**9.00±0.04**	**6.63±0.02**

Table 2: Evaluation results per domain and setting for **A**ction and **F**luent observations. Columns are as in Table 1.

Finally, we recognize that *Obra Dinn*, and information storygames in general, invite players to engage in a form of *online* plan recognition, that interleaves reasoning and exploration in a manner comparable to automated planning and execution. We focus here only on an *offline* model to establish a baseline, but future work will look to develop an online variant of the reasoning we have sketched herein.

Conclusion

For plan recognition to be up to the task of modeling the sensemaking requisite to playing information storygames, it needs to be capable of handling the kind of information intrinsic to them: non-linearly discovered ambiguous information. In this paper, we presented a modest step toward that: crisp definitions for partially-ordered and optional observations of fluents and actions, and what it means to satisfy each. We then demonstrated a novel compilation of this information that permits satisfactorily computing recognized plans. While we focus on predicting rational agent behavior, our future efforts will extend this work to cover boundedly-rational agents as well.

Ultimately, we aim to characterize what is mechanistically needed to simulate information storygame sensemaking. This work is a step toward formalizing the construction of mental and event models from plan-based first principles.

Acknowledgements

This material is based on work supported by the United States National Science Foundation (Grant #2046294). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States National Science Foundation. We thank our anonymous reviewers for their helpful feedback during peer review.

References

Aineto, D.; Jimenez, S.; and Onaindia, E. 2020. Observation Decoding with Sensor Models: Recognition Tasks via Classical Planning. *Proceedings of the International Conference on Automated Planning and Scheduling*, 30(1): 11–19.

Carberry, S. 1990. *Plan Recognition in Natural Language Dialogue*. MIT press.

Cardona-Rivera, R. E.; Jhala, A.; Porteous, J.; and Young, R. M. 2024. The Story So Far on Narrative Planning. In *Proceedings of the 34th International Conference on Automated Planning and Scheduling*, 489–499.

Cardona-Rivera, R. E.; and Li, B. 2016. PLOTSHOT: Generating Discourse-constrained Stories around Photos. In *Proceedings of the 12th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2–8.

Cardona-Rivera, R. E.; and Young, R. M. 2015. Symbolic Plan Recognition in Interactive Narrative Environments. In *Proceedings of the Joint Workshop on Intelligent Narrative Technologies and Social Believability in Games at the 11th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 16–22.

Cardona-Rivera, R. E.; and Young, R. M. 2019. Desiderata for a Computational Model of Human Online Narrative Sensemaking. In *the Working Notes of the 2019 AAAI Spring Symposium on Story-enabled Intelligence*.

Cook, M. 2020. Procedural Generation and Information Games. In *Proceedings of the 2020 IEEE Conference on Games*, 253–260. IEEE.

Farrell, R.; and Ware, S. 2020. Narrative Planning for Belief and Intention Recognition. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 16(1): 52–58.

Fikes, R. E.; and Nilsson, N. J. 1971. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, 2(3-4): 189–208.

Francis, T. 2019. Design Talk: Information Games. https://youtu.be/LFMEmS4PN00. Accessed: 2025-05-31.

Galvane, Q.; Ronfard, R.; Christie, M.; and Szilas, N. 2014. Narrative-driven Camera Control for Cinematic Replay of Computer Games. In *Proceedings of the 7th International Conference on Motion in Games*, 109–117.

Helmert, M. 2006. The Fast Downward Planning System. *Journal of Artificial Intelligence Research*, 26: 191–246.

Herzig, R.; Raboh, M.; Chechik, G.; Berant, J.; and Globerson, A. 2018. Mapping Images to Scene Graphs with Permutation-invariant Structured Prediction. In *Proceedings of the 32nd Conference on Advances in Neural Information Processing Systems*.

Hoffmann, J. 2003. The Metric-FF Planning System: Translating "Ignoring Delete Lists" to Numeric State Variables. *Journal of Artificial Intelligence Research*, 20: 291–341.

Johnson, J.; Krishna, R.; Stark, M.; Li, L.-J.; Shamma, D.; Bernstein, M.; and Fei-Fei, L. 2015. Image Retrieval Using Scene Graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3668–3678.

Krishna, R.; Zhu, Y.; Groth, O.; Johnson, J.; Hata, K.; Kravitz, J.; Chen, S.; Kalantidis, Y.; Li, L.; Shamma, D. A.; Bernstein, M. S.; and Fei-Fei, L. 2016. Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations. *CoRR*, abs/1602.07332.

Louarn, A.; Christie, M.; and Lamarche, F. 2018. Automated Staging for Virtual Cinematography. In *Proceedings of the 11th ACM SIGGRAPH Conference on Motion, Interaction and Games*, 1–10.

Martens, C.; Cardona-Rivera, R. E.; and Cohn, N. 2020. The Visual Narrative Engine: A Computational Model of the Visual Narrative Parallel Architecture. In *Proceedings of the 8th Annual Conference on Advances in Cognitive Systems*.

Masters, P.; and Sardina, S. 2019. Goal Recognition for Rational and Irrational Agents. In *Proceedings of the 18th International Conference on Autonomous Agents and Multiagent Systems*, 440–448.

McDermott, D.; Ghallab, M.; Howe, A.; Knoblock, C.; Ram, A.; Veloso, M.; Weld, D.; and Wilkins, D. E. 1998. PDDL–The Planning Domain Definition Language. Technical Report CVC TR-98-003/DCS TR-1165, Yale Center for Computational Vision and Control, New Haven, CT, USA.

Min, W.; Mott, B.; Rowe, J.; Taylor, R.; Wiebe, E.; Boyer, K.; and Lester, J. 2017. Multimodal Goal Recognition in Open-world Digital Games. In *Proceedings of the 13th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 80–86.

Pereira, R. F.; and Meneguzzi, F. 2017. Goal and plan recognition datasets using classical planning domains. https://zenodo.org/records/825878. Accessed: 2025-08-29.

Pope, L. 2018. *Return of the Obra Dinn*. 3903 LLC. Video game published for various platforms.

Poppe, R. 2010. A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6): 976–990.

Radvansky, G. A.; and Zacks, J. M. 2017. Event Boundaries in Memory and Cognition. *Current Opinion in Behavioral Sciences*, 17: 133–140.

Ramírez, M.; and Geffner, H. 2009. Plan Recognition as planning. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*.

Ramírez, M.; and Geffner, H. 2010. Probabilistic plan recognition using off-the-shelf classical planners. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*.

Shvo, M.; Klassen, T. Q.; Sohrabi, S.; and McIlraith, S. A. 2020. Epistemic Plan Recognition. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '20, 1251–1259. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450375184.

Siler, C.; and Ware, S. G. 2023. Knowledge Goal Recognition for Interactive Narratives. In *Proceedings of the Experimental AI in Games Workshop at the 19th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*.

Sohrabi, S.; Riabov, A. V.; and Udrea, O. 2016. Plan Recognition as Planning Revisited. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*.

Stewart, B. 2015. Environmental Storytelling. https://www.gamedeveloper.com/design/environmental-storytelling. Accessed: 2025-05-31.

Sukthankar, G.; Geib, C.; Bui, H. H.; Pynadath, D.; and Goldman, R. P. 2014. *Plan, Activity, and Intent Recognition: Theory and Practice*. Morgan Kaufmann.

Vered, M.; Kaminka, G. A.; and Biham, S. 2016. Online Goal Recognition through Mirroring: Humans and Agents. In *Proceedings of the 4th Annual Conference on Advances in Cognitive Systems*.