Error Tolerant Plan Recognition: An Empirical Investigation

Swaroop S. Vattam[1], David W. Aha[2], and Michael W. Floyd[3]

[1]NRC Postdoctoral Fellow; Naval Research Laboratory (Code 5514); Washington, DC, USA
[2]Naval Research Laboratory (Code 5514); Washington, DC, USA
[3]Knexus Research Corporation, Springfield, VA, USA
{swaroop.vattam.ctr.in, david.aha}@nrl.navy.mil | michael.floyd@knexusresearch.com

Abstract

Few plan recognition algorithms are designed to tolerate input errors. We describe a case-based plan recognition algorithm (SET-PR) that is robust to two input error types: missing and noisy actions. We extend our earlier work on SET-PR with more extensive evaluations by testing the utility of its novel action-sequence representation for plans and also investigate other design decisions (e.g., choice of similarity metric). We found that SET-PR outperformed a baseline algorithm for its ability to tolerate input errors, and that storing and leveraging state information in its plan representation substantially increases its performance.

1. Introduction

We are developing an intelligent agent to control a robot in joint human-robot team missions. This robot perceives the actions of its human teammates, recognizes their plans and goals, and then selects its actions accordingly. Our plan recognizer must operate on action information perceived by lower-level perception that is prone to *errors* (i.e., mislabeled and/or missing actions in the observed action sequences). Thus, error tolerance is a key design concern for plan recognition.

We describe the *Single-agent Error-Tolerant Plan Recognizer* (SET-PR), a case-based algorithm. Plan recognition algorithms typically employ a model or library of plans to recognize an ongoing plan from observed action sequences. SET-PR's plan representation (*action-sequence graphs*) encodes (1) knowledge about actions performed by an observed agent, as is normally done, and (2) the subsequent state. That is, plans in SET-PR's plan library contain action-state sequences rather than only action sequences. To process these, SET-PR performs graph matching to retrieve candidate plans, and thus must compute similarity efficiently. Degree sequence similarity metrics (e.g., Johnson, 1985; Bunke and Shearer 1998; Wallis et al. 2001) can be used for this task, but it is not clear which is preferable.

In §2 and §3, we describe related work and SET-PR, respectively. We introduced SET-PR in (Vattam et al. 2014) and reported its performance at varying levels of input error. In §4 we extend our empirical study by comparing SET-PR's ability to tolerate input errors vs. baselines, and studying how its plan representation and choice of similarity function influences its ability to tolerate errors. We found support for our hypotheses that SET-PR's action-sequence graph representation for plans and the inclusion of state information in these representations increases plan recognition performance in the presence of input errors. Finally, we discuss these results in §5 and provide concluding remarks in §6.

2. Related Work

Several plan recognition algorithms (Sukthankar et al. 2014) have used consistency-based (e.g., Hong 2001; Kautz and Allen, 1986; Lesh and Etzioni 1996) or probabilistic (e.g., Bui, 2003; Charniak and Goldman, 1991; Goldman et al. 1999; Pynadath and Wellman 2000) approaches. SET-PR exemplifies a less-studied third approach, namely case-based plan recognition (CBPR) (Cox and Kerkez 2006; Tecuci and Porter 2009). Some CBPR algorithms can work with incomplete plan libraries, incrementally learn plans, or respond to novel inputs outside the scope of their plan library using plan adaptation techniques. However, to our knowledge none have been designed for error-prone inputs, which is our focus.

Cox and Kerkez (2006) proposed a novel representation for storing and organizing plans in a plan library, based on action-state pairs and abstract states. It counts the number of instances of each type of generalized state predicate. SET-PR uses a similar representation, but stores and processes plans in an action-sequence graph. As a result, our similarity metrics also operate on graphs. Our encoding was inspired by *planning encoding graphs* (Serina 2010). Although there are syntactic similarities among these two types of graphs, important semantic differences exist; Serina's graphs encode a planning problem while ours encode a solution (i.e., a grounded plan).

Recently, Maynord et al. (2015) integrated SET-PR with hierarchical clustering techniques to increase its retrieval speed. Sánchez-Ruiz and Ontañón (2014) instead use Least Common Subsumer (LCS) Trees for this purpose. In this

paper, we focus on SET-PR's ability to tolerate input errors rather than methods for increasing its retrieval speed.

3. SET-PR

When our agent receives a set of observations, it invokes SET-PR to obtain a hypothesized plan for the observed agents. SET-PR is given a plan library C (i.e., a set of cases), where a *case* is a tuple $c = (\pi_0, g_0)$, π_0 is a (grounded) plan, and g_0 is a goal that is satisfied by π_0's execution.

Each plan is represented as an *action-state sequence* $\mathbb{s} = \langle(\boldsymbol{a_0}, \boldsymbol{s_0}), \dots, (\boldsymbol{a_n}, \boldsymbol{s_n})\rangle$, where each action $\boldsymbol{a_i}$ is a ground instance of an operator in the planning domain, and $\boldsymbol{s_i}$ is the state obtained by executing $\boldsymbol{a_i}$ in $\boldsymbol{s_{i-1}}$. We represent an action $\boldsymbol{a}$ in $(\boldsymbol{a}, \boldsymbol{s}) \in \mathbb{s}$ as a ground predication $\boldsymbol{p} = p(o_1{:}\, t_1, \dots, o_n{:}\, t_n)$, where $p \in \boldsymbol{P}$ (a finite set of predicate symbols), $o_i \in \boldsymbol{O}$ (a finite set of typed constants representing objects), and t_i is an instance of o_i (e.g., `stack(block:A, block:B)`, `on(block:A, block:B)`). A state $\boldsymbol{s}$ in $(\boldsymbol{a}, \boldsymbol{s}) \in \mathbb{s}$ is as a set of facts $\{\boldsymbol{p_1}, \boldsymbol{p_2}, \cdots\}$, where each $\boldsymbol{p_i}$ is a predication.

Inputs to SET-PR are also represented as action-state sequences.

3.1 Action-Sequence Graphs

SET-PR uses action-sequence graphs to represent action-state sequences. A *labeled directed graph* G is a 3-tuple $G = (V, E, \lambda)$, where V is a set of vertices, $E \subseteq V \times V$ is a set of edges, and $\lambda{:}\, V \cup E \rightarrow \wp_s(L)$ assigns labels to vertices and edges. Here, an edge $e = [v, u] \in E$ is directed from v to u, where v is the edge's *source* node and u is the *target* node; L is a finite set of symbolic labels; and $\wp_s(L)$, a set of all the *multisets* on L, permits multiple non-unique labels for a node or an edge (for properties of $\wp_s(L)$ please see Serina (2010)).

The *union* $G_1 \cup G_2$ of two graphs $G_1 = (V_1, E_1, \lambda_1)$ and $G_2 = (V_2, E_2, \lambda_2)$ is the graph $G = (V, E, \lambda)$, where $V = V_1 \cup V_2$, $E = E_1 \cup E_2$, and

$$\lambda(x) = \begin{cases} \lambda_1(x), if\ x \in (V_1 \setminus V_2) \vee x \in (E_1 \setminus E_2) \\ \lambda_2(x), if\ x \in (V_2 \setminus V_1) \vee x \in (E_2 \setminus E_1) \\ \lambda_1(x) \cup \lambda_2(x), otherwise \end{cases}$$

Definition: Given ground atom $\boldsymbol{p}$ representing an action $\boldsymbol{a}$ or a fact of state $\boldsymbol{s}$ in the k^{th} action-state pair $(\boldsymbol{a}, \boldsymbol{s})_k \in \mathbb{s}$, a *predicate encoding graph* is a labeled directed graph $\mathcal{E}^p(\boldsymbol{p}) = (V_p, E_p, \lambda_p)$ where:

- $V_p = \begin{cases} \{A_{k_p}, o_1, \dots, o_n\}, \text{if } \boldsymbol{p} \text{ is an action} \\ \{S_{k_p}, o_1, \dots, o_n\}, \text{if } \boldsymbol{p} \text{ is a state fact} \end{cases}$
- $E_p = \begin{cases} [A_{k_p}, o_1] \cup \bigcup_{i=1,n-1;j=i+1,n}[o_i, o_j], \text{if } \boldsymbol{p} \text{ is an action} \\ [S_{k_p}, o_1] \cup \bigcup_{i=1,n-1;j=i+1,n}[o_i, o_j], \text{if } \boldsymbol{p} \text{ is a state fact} \end{cases}$
- $\lambda_p(A_{k_p}) = \{A_{k_p}\};\ \lambda_p(S_{k_p}) = \{S_{k_p}\};\ \lambda_p(o_i) = \{t_i\}$ for $i = 1, \dots, n$
- $\lambda_p([A_{k_p}, o_1]) = \{A_{k_p}^{0,1}\};\ \lambda_p([S_{k_p}, o_1]) = \{S_{k_p}^{0,1}\};$ $\forall [o_i, o_j] \in E_p,$
$$\lambda_p([o_i, o_j]) = \begin{cases} \{A_{k_p}^{i,j}\}, \text{if } \boldsymbol{p} \text{ is an action} \\ \{S_{k_p}^{i,j}\}, \text{if } \boldsymbol{p} \text{ is a state fact} \end{cases}$$

As an interpretation of this definition suppose we have a predication $\boldsymbol{p} = p(o_1{:}\, t_1, \dots, o_n{:}\, t_n)$. Depending on whether $\boldsymbol{p}$ represents an action or a state fact, the first node of the predicate encoding graph $\mathcal{E}^p(\boldsymbol{p})$ is either A_{k_p} or S_{k_p} (labeled $\{A_{k_p}\}$ or $\{S_{k_p}\}$). Suppose it is an action predicate. A_{k_p} is then connected to the second node of this graph, the object node o_1 (labeled $\{t_1\}$), through the edge $[A_{k_p}, o_1]$ (labeled $\{A_{k_p}^{0,1}\}$). Next, o_1 is connected to the third node o_2 (labeled $\{t_2\}$) through the edge $[o_1, o_2]$ (labeled $\{A_{k_p}^{1,2}\}$), then to the fourth node o_3 (labeled $\{t_3\}$) through the edge $[o_1, o_3]$ (labeled $\{A_{k_p}^{1,3}\}$), and so on. Suppose also the third node o_2 is connected to o_3 through $A_{k_p}^{2,3}$, to o_4 through $A_{k_p}^{2,4}$, with appropriate labels, and so on.

Definition: An *action-sequence graph* of an action-state sequence $\mathbb{s}$ is a labeled directed graph $\mathcal{E}^{\mathbb{s}} = \bigcup_{(\boldsymbol{a},\boldsymbol{s})\in\mathbb{s}}(\mathcal{E}(\boldsymbol{a}) \cup \bigcup_{\boldsymbol{p}\in\boldsymbol{s}} \mathcal{E}(\boldsymbol{p}))$, a union of the predicate encoding graphs of the actions and state facts in $\mathbb{s}$.

Space constraints prevent providing more detail. Please see (Vattam et al. 2014) for examples of action-sequence graphs and their construction from action-state sequences.

3.2 Case Retrieval

SET-PR matches an input action-sequence graph $\mathbb{s}^{\boldsymbol{target}}$ with plans in the cases of C. The case $\boldsymbol{c} = (\boldsymbol{\pi_0}, \boldsymbol{g_0})$ whose plan $\boldsymbol{c}.\boldsymbol{\pi_0}$ is most similar is retrieved as the recognized plan, and $\boldsymbol{c}.\boldsymbol{g_0}$ is the recognized goal.

To match graphs, we compute their maximum common subgraph (MCS). Computing the MCS between two or more graphs is NP-Complete, restricting applicability to only small plan recognition problems. Alternatively, many approximate graph similarity measures exist. One class of such similarity metrics, based on graph *degree sequences*, has been used successfully to match chemical structures (Raymond and Willett 2002).

Below, we describe four degree sequence similarity metrics that we will test in SET-PR. These metrics, denoted as sim_{str}, compute plan similarity based on the approximate structural similarity of their graph representations.

Let G_1 and G_2 be the two action-sequence graphs being compared. First, the set of vertices in each graph is divided into l partitions by label type, and then sorted in a non-

increasing total order by degree[1]. Let L_1^i and L_2^i denote the sorted degree sequences of a partition i in the action-sequence graphs G_1 and G_2, respectively. An upper bound on the number of vertices $V(G_1, G_2)$ and edges $E(G_1, G_2)$ of the MCS of these two graphs can then be computed as:

$$\left|\text{mcs}(G_1, G_2)\right| = V(G_1, G_2) + E(G_1, G_2), \text{ where}$$

$$V(G_1, G_2) = \sum_{i=1}^{l} min\left(\left|L_1^i\right|, \left|L_2^i\right|\right)$$

$$E(G_1, G_2) = \left\lfloor \sum_{i=1}^{l} \sum_{j=1}^{min\left(\left|L_1^i\right|, \left|L_2^i\right|\right)} \frac{min\left(\left|E\left(v_1^{i,j}\right)\right|, \left|E\left(v_2^{i,j}\right)\right|\right)}{2} \right\rfloor$$

where $v_1^{i,j}$ denotes the j^{th} vertex of the L_1^i sorted degree sequence, and $E\left(v_1^{i,j}\right)$ denotes the set of edges connected to vertex $v_1^{i,j}$.

We consider the following four similarity metrics, which are variations on the above properties.

- **J** Johnson (Johnson 1985):

$$sim_{str}(G_1, G_2) = \frac{\left(\left|\text{mcs}(G_1, G_2)\right|\right)^2}{|G_1| \cdot |G_2|}$$

- **B** Bunke (Bunke and Shearer 1998):

$$sim_{str}(G_1, G_2) = \frac{\left(\left|\text{mcs}(G_1, G_2)\right|\right)^2}{\max(|G_1|, |G_2|)}$$

- **W** Wallis (Wallis et al. 2001):

$$sim_{str}(G_1, G_2) = \frac{\left(\left|\text{mcs}(G_1, G_2)\right|\right)^2}{|G_1| + |G_2| - \left|\text{mcs}(G_1, G_2)\right|}$$

- **S** Simpson (Ellis et al. 1993):

$$sim_{str}(G_1, G_2) = \frac{\left(\left|\text{mcs}(G_1, G_2)\right|\right)^2}{\min(|G_1|, |G_2|)}$$

Two plans that are similar in structure can differ drastically in semantics. For instance, a plan to travel to a grocery store to buy milk might coincidentally be structurally similar to a plan to travel to the airport to receive a visitor. To mitigate this issue, we use a weighted combination of structural similarity and semantic similarity, denoted as sim_{obj}, as our final similarity metric:

$$\text{sim}(G_1, G_2) = \alpha\, \text{sim}_{str}(G_1, G_2) + (1 - \alpha)\text{sim}_{obj}(G_1, G_2),$$

where $\text{sim}_{obj}(G_1, G_2) = \frac{O_{\mathbb{S}} \cap O_{\pi_i}}{O_{\mathbb{S}} \cup O_{\pi_i}}$ is the Jaccard coefficient of the set of (grounded) objects in G_1 and G_2, and α $(0 \leq \alpha \leq 1)$ governs the weights associated with sim_{str} and sim_{obj}.

4. Empirical Study

Our empirical study builds on our earlier pilot study (Vattam et al. 2014), where we tested SET-PR at varying input error levels but did not compare it to a baseline. Also we did not compare different variants of SET-PR. In this study, we investigated the following hypotheses:

H1: SET-PR's action-sequence graph representation for plans increases recognition performance in the presence of input errors.

H2: Including state information in input action sequences and plans improves error tolerance.

H3: Combining structural and semantic similarity outperforms using either in isolation.

3.1 Empirical Method

We compared the performance of a baseline algorithm with three versions of SET-PR, all using **J** for graph matching. (We consider the other similarity metrics in §4.)

- **Baseline**: Inputs and plans contained ⟨action⟩ sequences (no state information), treated as symbols (not converted to a graph representation); matching was computed using *edit distance* (no graph matching).
- **SET-PR[A,0.5]**: Inputs and plans contained ⟨action⟩ sequences (no state information), represented as action-sequence graphs; $\alpha = 0.5$ (equal weights for structural and semantic similarity).
- **SET-PR[AS,0.5]**: Inputs and plans contained ⟨action, state⟩ sequences, represented as action-sequence graphs; $\alpha = 0.5$.
- **SET-PR[AS,0.33]**: This is a variant in which $\alpha = 0.33$ (slightly lower weight for structural similarity).

We conducted our experiments in the paradigmatic blocks world domain because it is simple and permits the quick automatic generation of a plan library with the desired characteristics. We used the hierarchal task network (HTN) planner SHOP2 (Nau et al. 2003) to generate plans for our library. Planning problems were created by randomly selecting initial and goal states (ensuring that the goal can be reached from the initial state), and given as input to SHOP2. The number of blocks used to generate the plans ranged from 9 to 12. We used this method to generate 100 plans for our library. The average plan length was 12.48. In the baseline condition, we stored the generated plan along with the goal as a case in the case base. In the non-baseline conditions, the generated plan was converted into an action-sequence graph (using actions only in SET-PR[A,0.5], and using actions and states in SET-PR[AS,0.5/0.33]), and stored along with the goal as a case in the case base.

We used the following plan recognition metrics (Blaylock and Allen 2005): (1) precision, (2) convergence rate, and (3) convergence point. To understand these metrics, consider a plan recognition session in which the recognizer is given x input actions, which are streamed sequentially. After observing each action, the recognizer uses the available

[1]The degree of a vertex v of a graph is the number of edges that touch v.

action sequence to query and predict a plan. The first query will be $\langle a_1 \rangle$, the second $\langle a_1, a_2 \rangle$, and so on until $\langle a_1, a_2, \cdots a_x \rangle$ (in SET-PR[AS] these will be action-state sequences). Each session consists of x queries and predictions. *Precision* reports the number of correct predictions divided by total predictions for a single session.[2] *Convergence* is true if the correct plan is predicted by the end of the session and false otherwise. If a correct prediction is followed by an incorrect prediction at any point in the observation session, the convergence flag will be reset to false. *Convergence rate* is the percentage of sessions that converged to true. If a session converges, *convergence point* reports the number of actions after which the session converged to the correct plan divided by the total number of actions. Convergence point is averaged only for those sessions that converge. Lower values for convergence point indicate better performance, whereas higher values for convergence rate and precision indicate better performance.

We evaluated the plan recognition metrics using the leave-one-in (Aha and Breslow 1997) testing strategy as follows. For each randomly selected case $c = (\pi_0, g_0) \in \mathsf{C}$, we copied plan π, randomly distorted its action-sequence $\langle (null, s_0), (a_1, s_1), \ldots, (a_g, s_g) \rangle$ to introduce a fixed and equal amount of mislabeled and missing error (for mislabeled, a specified percentage of actions in π was randomly chosen, and each was replaced with another action randomly chosen from the domain; for missing, a specified percentage of actions was randomly chosen, and each was replaced with an unidentified marker '*'.). This distorted plan was used as an incremental query to SET-PR (i.e., initially with only its first ⟨action, state⟩ pair, and then repeatedly adding the next such pair in its sequence). The error levels tested were {10%, 20%, 30%, …, 90%}.

3.2 Results

Figure 1 plots performance versus error levels across three metrics. For convergence rate, SET-PR[AS,0.5] and SET-PR[AS,0.33] outperformed Baseline and SET-PR[A,0.5]. Baseline's convergence fell sharply between 10% and 20% error rate, while SET-PR[A,0.5]'s degradation was more gradual, though it reached low levels at 40%. SET-PR[AS,0.5] and SET-PR[AS,0.33] maintained a convergence rate of 35% to 50% even at higher error rates.

For average precision, in the absence of any error, Baseline's precision was higher than the SET-PR variants. This is because SET-PR's approximate graph matching technique used can assign the same score to multiple plans with minor differences, in which case a random plan was selected. This can reduce average precision. With greater plan diversity in the library, we conjecture the performance of SET-PR will be similar to that of the Baseline in the zero

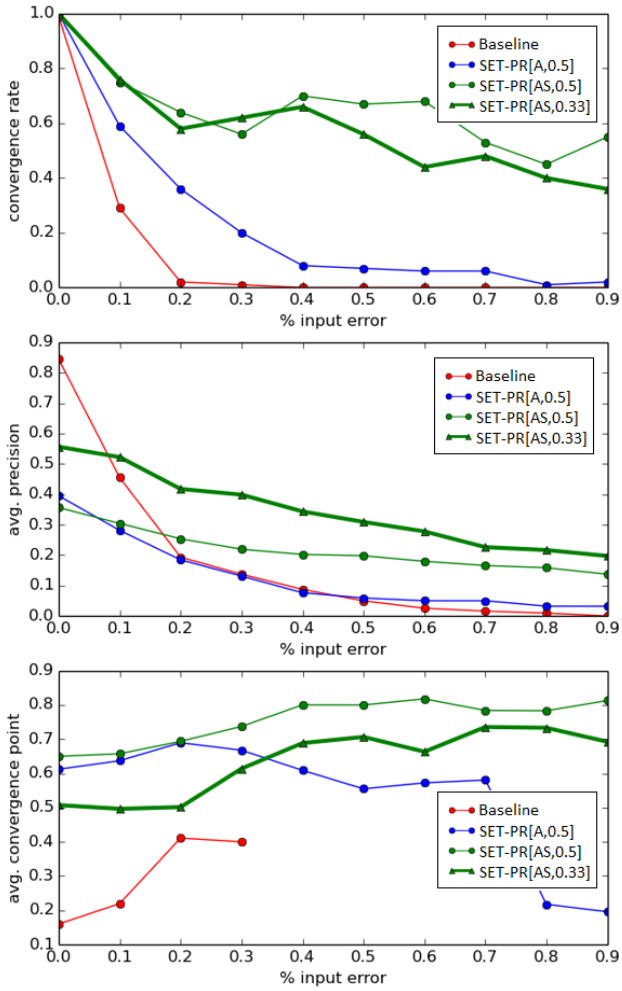

Figure 1*:* Performance of Baseline and three variations of SET-PR with a varying input error rate using three metrics.

error case. However, in the presence of error Baseline's precision fell sharply. Again, SET-PR[AS,0.5] and SET-PR[AS,0.33] performed best for higher error levels.

For average convergence point, Baseline recorded the lowest values, but that is not indicative of its superior performance because its convergence rate is low even in the 10% input error condition (the converged set had too few data points). Similarly, the convergence point for SET-PR[A,0.5] does not afford meaningful comparison beyond the 20% error rate. Only SET-PR[AS,0.5] and SET-PR[AS,0.33] can be meaningfully compared in this test, and SET-PR[AS,0.33] performed better at all error rates.

For convergence rate and precision, Baseline performed comparatively poorly at all non-zero error levels, particularly when using action-state sequence representations in SET-PR, lending some support to **H1**. These results also lend support to **H2**; actions-only SET-

[2] This should not be confused with typical precision/recall definitions involving false positives and false negatives.

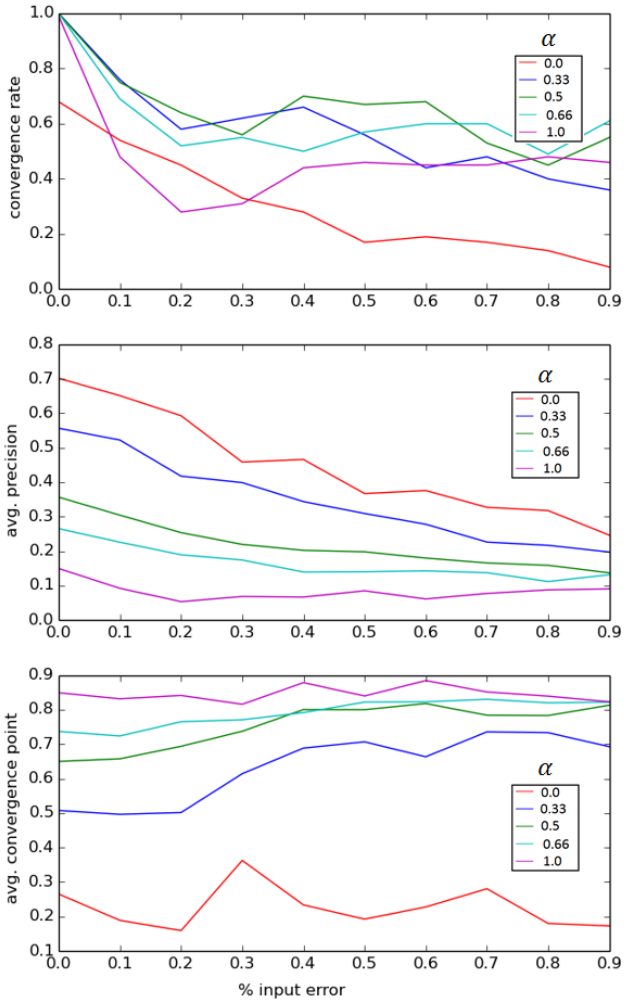

Figure 2*:* Performance of SET-PR[AS], varying from purely semantic ($\alpha = 0$) to purely structural ($\alpha = 1$) similarity.

PR[A,0.5] was outperformed by the other two variants with state information on all three metrics at all non-zero error levels (with the exception of convergence point at high error levels characterized by a low convergence rate).

Figure 2 displays the performance of SET-PR[AS] with levels of α ranging from 0.0 (purely semantic similarity) to 1.0 (purely structural) in four increments. This shows that using only semantic similarity performs well on two metrics but poorly on a third, while purely structural similarity performs poorly for two of the metrics. The best overall performance was attained when α = 0.33 taking, all three parameters into account. This lends support to **H3.**

4. Discussion

Our results for **H2** suggest that plan representations rich in state information, such as SET-PR's action-state sequences, enable more informed predictions because states capture the context of actions. Plan recognition techniques that rely

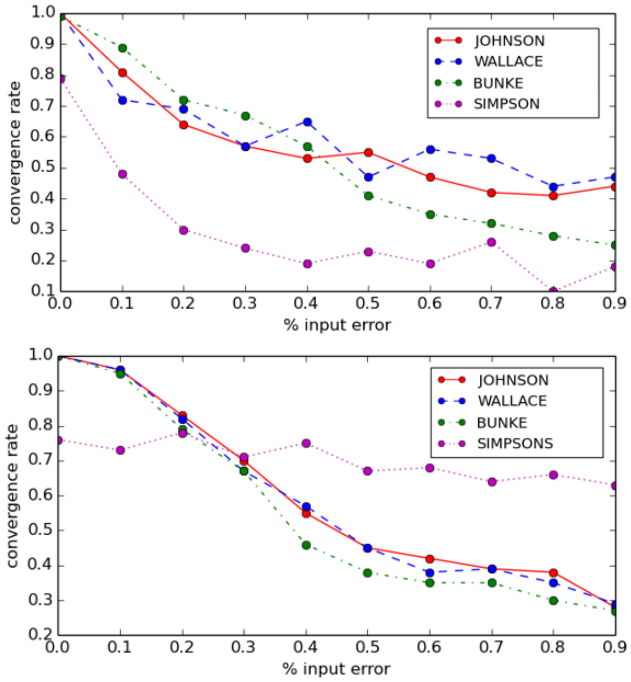

Figure 3*:* Convergence rate of SET-PR[AS,0.33] with different degree sequence similarity metrics, given (a) both mislabeled and missing actions, and (b) only missing actions.

solely on actions exhibit brittleness even when a small proportion of input actions are mislabeled or missing. Our results for **H3** suggests that SET-PR is sensitive to α (structural vs semantic similarity), but the best observed value of α = 0.33 could be domain specific. In future work with other domains, we will assess the extent to which SET-PR is sensitive to α.

Our study in §3 may be influenced by several factors (e.g., the similarity metric that was used). In an exploratory analysis, we examined whether **J** was an appropriate choice by testing all four similarity metrics using SET-PR[AS,0.33]. As Figure 3(a) shows for convergence rate, while performance deteriorates with increasing levels of input error, **J** performs on par with **W**, and outperformed **S** substantially. Similarly, we found that **J**, **W**, and **B** outperformed **S** on precision, while **S** performed well on convergence point, though this is not indicative of its superior performance because its convergence rate was low beyond the 20% error rate (i.e., too few data points in the converged set to derive a trend). This suggests that, for these studies, **J** is an appropriate choice.

However, a factorial study of other design choices would reveal a more complicated story. For example, Figure 3(b) plots the convergence rate for the same algorithms but with input errors containing only one type of error (missing actions). In contrast to using the other similarity metrics, the convergence rate of SET-PR[AS,0.33] for **S** does not deteriorate with higher error levels. Our conjecture is that **S** is more sensitive to the size of the MCS, and theoretical

analyses may reveal that the MCS (as a percentage of graph size) of two randomly-sampled graphs, for higher error rates, is much higher when the errors are constrained to missing actions. We will test for this in future work, and whether this behavior is limited to our current domain and plan libraries.

5. Summary and Future Work

We described SET-PR, a case-based approach to the problem of plan recognition that can tolerate mislabeled and missing actions in the input action sequences. We highlighted SET-PR's case representation (*action-sequence graphs*) and SET-PR's similarity function, which combines degree sequences similarity and semantic similarity for matching action-sequence graphs. We described an empirical study where we found evidence to support our hypotheses that SET-PR's action-sequence graph representation for plans and the inclusion of state information in these representations increases plan recognition performance in the presence of input errors. We also found that combining structural and semantic similarity outperforms using either in isolation.

For future work, we will conduct a factorial study of our design choices, with the objective of explaining some of the trends that we observed (e.g., why SET-PR[AS,0.33] with Simpson's similarity metric maintained a high convergence rate even at higher levels of errors). We will also integrate and test SET-PR in other domains, including a human-robot teaming domain. We will also compare SET-PR's performance with that of other state-of-the-art plan recognizers in the presence of input errors. Finally, we will examine more sophisticated graph similarity metrics (e.g., graph kernels) and compare them against the simple degree sequence metrics that SET-PR currently uses.

Acknowledgements

Thanks to OSD ASD (R&E) for sponsoring this research. The views and opinions in this paper are those of the authors and should not be interpreted as representing the official views or policies of NRL or OSD.

References

Aha, D.W. & Breslow, L.A. (1997). Refining conversational case libraries. *Proceedings of the Second International Conference on CBR* (267–278). Providence, RI: Springer.

Blaylock, N., & Allen, J. (2005). Recognizing instantiated goals using statistical methods. In G. Kaminka, D.V. Pynadath, & C.W. Geib (Eds.) *Modeling others from observations: Papers from the IJCAI Workshop*.

Bui, H. (2003). A general model for online probabilistic plan recognition. *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence* (pp. 1309–1315). Acapulco, Mexico: Morgan Kaufmann.

Bunke, H., & Shearer, K. (1998). A graph distance metric based on the maximum common subgraph. *Pattern Recognition*, *19*(3), 255-259.

Charniak, E., & Goldman, R. (1991). A probabilistic model of plan recognition. *Proceedings of the 9th National Conference on AI* (pp. 160-165). Anaheim, CA: AAAI Press.

Cox, M. T., & Kerkez, B. (2006). Case-based plan recognition with novel input. *Control and Intelligent Systems*, *34*(2), 96-104.

Ellis, D., Furner-Hines, J., & Willett, P. (1993). Measuring the degree of similarity between objects in text retrieval systems. *Perspectives in Information Management*, *3*(2), 128-149.

Goldman, R.P., Geib, C.W., & Miller, C.A. (1999). A new model of plan recognition. *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence* (pp. 245-254). Bled, Slovenia: Morgan Kaufmann.

Hong, J. (2001). Goal recognition through goal graph analysis. *Journal of Artificial Intelligence Research*, *15*, 1-30.

Johnson, M. (1985). *Relating metrics, lines and variables defined on graphs to problems in medicinal chemistry*. NY: Wiley.

Kautz, H., & Allen, J. (1986). Generalized plan recognition. *Proceedings of the Fifth National Conference on Artificial Intelligence* (pp. 32-37). Philadelphia, PA: Morgan Kaufmann.

Lesh, N., & Etzioni, O. (1996). Scaling up goal recognition. *Proceedings of the Fifth International Conference on Principles of Knowledge Representation and Reasoning* (pp. 178-189). Cambridge, MA: Morgan Kaufmann.

Maynord, M., Vattam, S. & Aha, D.W. (2015). Increasing the runtime speed of case-based plan recognition. In *Proceedings of the 28th FLAIRS Conference*. Hollywood, FL: AAAI Press.

Nau, D.S., Au, T.C., Ilghami, O., Kuter, U., Murdock, J.W., Wu, D., & Yaman, F. (2003). SHOP2: An HTN planning system. *Journal of Artificial Intelligence Research*, *20*, 379-404.

Pynadath, D.V., & Wellman, M.P. (2000). Probabilistic state-dependent grammars for plan recognition. *Proceedings of the Conference on Uncertainty in Artificial Intelligence* (pp. 507–514). San Francisco, CA: Morgan Kaufmann.

Raymond, J. W., & Willett, P. (2002). Maximum common subgraph isomorphism algorithms for the matching of chemical structures. *Journal of Computer-Aided Molecular Design*, *16*, 521–533.

Sánchez-Ruiz, A.A., & Ontañón, S. (2014). Least common subsumer trees for plan retrieval. *Proceedings of the Twenty-Second International Conference on Case-Based Reasoning* (pp. 405-419). Cork, Ireland: Springer.

Serina, I. (2010). Kernel functions for case-based planning. *Artificial Intelligence*, *174*(16), 1369-1406.

Sukthankar, G., Goldman, R., Geib, C., Pynadath, D., Bui, H. (2014). An introduction to plan, activity, and intent recognition. In G. Sukthankar, R.P. Goldman, C. Geib, D.V. Pynadath, and H.H. Bui (Eds.) *Plan, Activity, and Intent Recognition*. Philadelphia, PA: Elsevier.

Tecuci, D., & Porter, B.W. (2009). Memory based goal schema recognition. In *Proceedings of the Twenty-Second International Florida Artificial Intelligence Research Society Conference*. Sanibel Island, FL: AAAI Press.

Vattam, S., Aha, D.W. & Floyd, M. (2014). Case-based plan recognition using action sequence graphs. *Proceedings of the Twenty-Second International Conference on Case-Based Reasoning* (pp. 495-510). Cork, Ireland: Springer.

Wallis, W. D., Shoubridge, P., Kraetz, M., & Ray, D. (2001). Graph distances using graph union. *Pattern Recognition Letters*, *22*(6), 701-704.