AI Inference of Team Effectiveness for Training and Operations

Robert Hyland, Kenneth R. Lu, Spencer K. Lynn, Stephen J. Marotta, James Niehaus, William Norsworthy Jr., Avi Pfeffer, Curtis Q. Wu, and Bryan Loyall

Charles River Analytics, Inc.

Cambridge, MA

[rhyland, klu, slynn, smarotta, jniehaus, wnorsworthy, apfeffer, cwu, bloyall]@cra.com

ABSTRACT

How can we build artificial intelligence (AI) that robustly recognizes how well a team is doing from behavioral data that exhibits the full range of human complexity and dynamics? One method is cognitive inversion. An AI with a causal model of human behavior that is sufficiently dynamic to account for behavioral variability and teammate interactivity and scoped to a set of tasks and interactions of interest, combined with a probabilistic program inference, can invert that behavioral model to generate hypotheses about the underlying goals and causes of observed behavior. As a tutor, coach, or teammate, the AI can then intervene to assist when needed. Here, we describe our prototype cognitive inversion system called Prescient, Socially Intelligent Coach (PSI-Coach) and its supporting components. PSI-Coach monitors team members to recognize their goals, mental states, and behaviors from dynamic streams of actions by combining probabilistic programming inference with a cognitive architecture designed to capture human variation. PSI-Coach uses those recognized cognitive states to infer a team's shared mental models and whether they are in alignment or skewed; analyze these goals, mental states, behaviors, and shared mental models to compute practical, real-time team performance indicators; and integrate all of this information with interactive-narrative technology to plan minimally intrusive, effective, strategically timed interventions that help to improve team performance. In experiments, we demonstrated the ability of cognitive inversion to automatically identify team process problems unique to different teams and their situation dynamics, and, based on those results, we show PSI-Coach's ability to provide timely, tailored intervention content that improved team processes. Cognitive inversion showed a 35% increase over rule-based comparison systems' real-time inferences ($p<0.05$), and led PSI-Coach to exhibit a 42%–68% increase in agreement with human coaches on interventions over a baseline inference method ($p<0.05$).

ABOUT THE AUTHORS

Robert Hyland is a Principal Scientist and Director of Program Transition at Charles River Analytics. His areas of interest include artificial intelligence, behavioral models, and mixed-initiative decision support.

Kenneth R. Lu is a Scientist at Charles River Analytics. His areas of research include program analysis, probabilistic reasoning, and machine learning, along with their applications in building fault-tolerant and adaptive software.

Spencer Lynn, Ph.D., is a Senior Scientist at Charles River Analytics. His research uses principles of behavioral ecology and neuroscience to create biologically inspired cognitive architectures and autonomous agents.

Stephen J. Marotta is a Senior Software Engineer at Charles River Analytics. His interests include integration of intelligent agents with simulation environments and interface designs for video analysis and situational awareness tools.

James Niehaus, Ph.D., is a Principal Scientist and Director at Charles River Analytics. His areas of expertise include artificial intelligence and cognitive models applied to explainable AI, adaptation to open-world novelty, and human–AI teaming.

William Norsworthy Jr., is a Software Engineer at Charles River Analytics. His interests include implementation of agent architectures and advanced typed ontology systems, interface design, and game development.

Avi Pfeffer, Ph.D., is Chief Scientist at Charles River Analytics. His research spans a variety of computational intelligence techniques including probabilistic reasoning, machine learning, and computational game theory.

Curtis Q. Wu is Chief Software Engineer at Charles River Analytics. His interests include self-adapting software, team performance optimization, resilient operating systems, and intelligent agent frameworks.

Bryan Loyall, Ph.D., is a Principal Scientist and Director of Technology Innovation at Charles River Analytics. His interests include integrated cognitive architectures, AI-based understanding of complex human behavior, and hybrid AI combining symbolic and data-driven technologies.

AI Inference of Team Effectiveness for Training and Operations

Robert Hyland, Kenneth R. Lu, Spencer K. Lynn, Stephen J. Marotta, James Niehaus, William Norsworthy Jr., Avi Pfeffer, Curtis Q. Wu, and Bryan Loyall

Charles River Analytics, Inc.

Cambridge, MA

[rhyland, klu, slynn, smarotta, jniehaus, wnorsworthy, apfeffer, cwu, bloyall]@cra.com

INTRODUCTION

Improving team performance requires deep understanding of what a team is doing—whether rational, irrational, or idiosyncratic—and administering effective, nonannoying interventions so the team can reach high levels of achievement. Recognizing detailed user goals, mental states, and behaviors (in all of their human complexity) from low-level actions in dynamic open worlds is a challenging task requiring real-time inference of complex, human cognitive processes. Predicting and planning well-timed, effective interventions requires a depth of understanding that takes human coaches years to learn and has repeatedly failed (in sometimes spectacular ways) in research and commercial artificial intelligence (AI) systems.

To meet these challenges, we created a Prescient, Socially Intelligent Coach (PSI-Coach). PSI-Coach is designed to unobtrusively monitor each team member to (1) recognize goals, mental states, and behaviors without the assumption of rationality by extending an expressive AI cognitive architecture with probabilistic programming languages (PPLs); (2) robustly recognize details of dynamic open-world behavior using inference over reactive cognitive architecture language features; (3) recognize aligned and skewed shared mental models within teams using joint behavior inference; (4) measure team performance indicators in real time using inferred mental states combined with novel, dynamic-task extensions to our partner's test of collective intelligence; and (5) maximize team performance using experience management algorithms that predict team behaviors and reason about both the efficacy and disruption of interventions. PSI-Coach makes these predictions and interventions using a planning technology that reasons about the effectiveness, timing, and disruption of potential interventions. This technology has delivered preliminary results, unobtrusively guiding interactive stories with unpredictable participants.

Here, we will focus on objective number 1, above: the inference of human teammate goals via a process we call *cognitive inversion.* The term derives from inverting a cognitive process model that, in the forward direction, generates goals and subgoals, eventually producing observable behavior in the world. Inverting that model ("running it backward") uses the observable behaviors—the model of how they can be generated by goals—and infers which goals were likely to have produced the observed behavior. Cognitive inversion, as described here, builds on decades of earlier work in expressive AI cognitive architectures for generating natural human behavior. For example, it had long been believed that computer vision could be performed by inverting the computer graphics pipeline (i.e., running it backward), but this concept only recently became practical with breakthroughs in PPLs (Kulkarni, et al., 2015). This general idea has also been applied to inverting physics simulators (Bates, et al., 2015) and seismic models (Arora, et al., 2013). We integrate recent breakthroughs in PPLs (Pfeffer, 2016) that allow generative models of behavior to be inverted for goal recognition.

The Current Study

An AI with a causal model of human behavior that is sufficiently dynamic to account for behavioral variability and teammate interactivity and scoped to a set of tasks and interactions of interest, combined with probabilistic program inference, can invert that behavioral model to generate hypotheses about the underlying goals and causes of observed behavior. A cognitive inverter, then, needs three things: observations to explain, a cognitive model to invert, and a probabilistic framework within which to infer likelihoods of causal paths linking goals to behaviors.

First, the cognitive inverter needs observable behavior inputs. We implemented a team-based search and rescue (SAR) task in a Minecraft™ testbed. We recorded the in-game behavior (actions) of teams of humans whose goal was to locate and triage casualties in a mass causality event. Example actions we recorded included movement (i.e., walking) within a building, entering rooms in the building, and marking casualties for levels of triage.

Second, the cognitive inverter needs a domain specific language (DSL) capable of generating the observed behavior. The DSL encodes goals, subgoals, and possible behavioral methods of achieving those goals. People are complex and dynamic; they do multiple things at the same time, switch between goals and tasks, pause activities and return to them later, and react to opportunities and changes in the world. A sufficiently expressive DSL should be capable of capturing this variation and also accommodate irrational behavior. An assumption of rationality pervades many AI systems, cognitive theories, and team theories, greatly restricting their understanding of human behavior.

Our system inverts an expressive cognitive model built on Charles River Analytics' Hap AI cognitive architecture (Sliva, 2016; Loyall et al., 2004; Loyall, 1997). Hap is a multiagent scripting language with syntax and computational architecture that performs parallel goal execution, prioritizes goals, and connects to simulator environments to sense and take actions (Figure 1). Hap has been designed over 30 years to create believable (including irrational), cognitively plausible, and socially realistic agents. Hap has spawned six research and commercial variants, which have been used as a foundation for dozens of research projects in socially engaging interactive agents (e.g., at Carnegie Mellon University; Stanford University; the Georgia Institute of Technology; and the University of California, Santa Cruz; as well as several research laboratories). Versions of Hap have been used as a foundation for over a dozen Ph.D. theses. Hap is currently on its sixth version, and AI agents based on Hap have been created for a diverse range of domains (e.g., modeling medical teams, squad-level AI Army tactics, fourth-generation combat pilots, characters in interactive stories, cyber social-attack-vector simulations, and distractible missile defense operators).

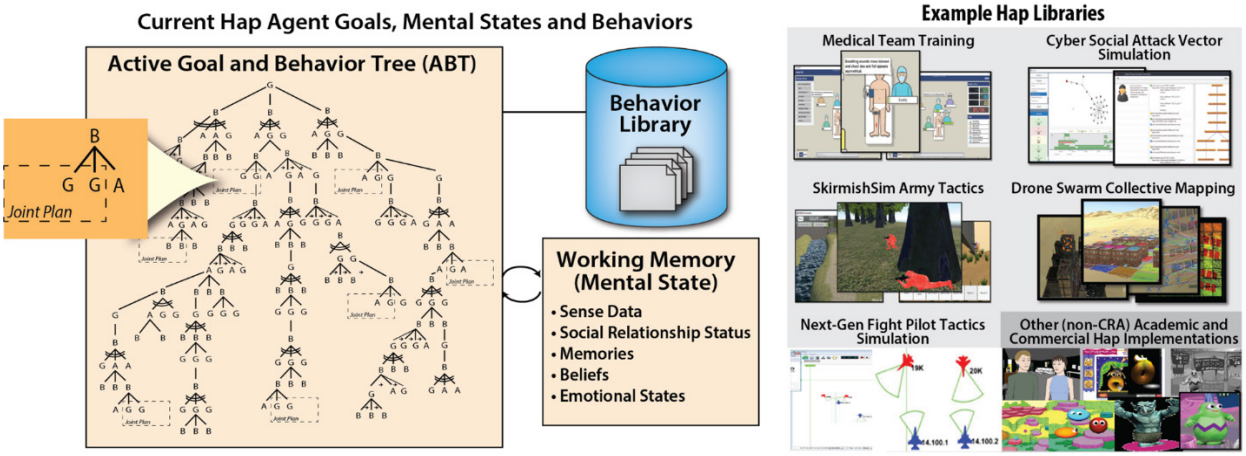

Figure 1. Hap AI cognitive architecture (left) and several example Hap applications (right).

By using Hap for recognition, the Cognition Inverter can recognize strategies as well as natural human idiosyncratic behaviors as a key foundation for assisting team performance. Example goals, strategies, and behaviors we encoded included search strategies, casualty triage and treatment strategies, disoriented wandering, revisiting of rooms or casualties, transporting casualties, and assisting teammates (e.g., helping to transport, removing obstacles).

Third, the cognitive inverter needs a PPL to construct and sample a probabilistic model of relationships among goals and behaviors expressed in the DSL. The constructed model represents hypothesized goal pathways leading to observed behavior. The sampling over structures composed in the DSL is the inversion step; it inverts the direction of the forward-running expressive cognitive model to infer the goals and mental states of team members from their behavior. The resulting probabilistic program synthesis results in a dynamic Bayesian network (DBN), the nodes of which are goals and subgoals, with edges linking causal paths from goal to subgoal to behavior. Overall, the DBN graph represents hypothetical goal pathways that explain the observations. The DBN nodes can be queried for their posterior beliefs—the evidentiary support that a particular goal had a causal role in generating observed behaviors.

We began research and development by manually constructing a DBN cognitive inverter. We built this initial DBN in the Figaro™ PPL (Pfeffer, 2016). Figaro is an open-source PPL (cra.com/projects/figaro/) developed at Charles River that expresses probabilistic models as data structures in the Scala programming language. We successful showed cognitive inversion with this DBN using the Minecraft data. Subsequently, we built the Hap DSL and the Inverse Hap cognitive inverter, iHap, using the Scruff™ PPL (Pfeffer, et al., 2021) to synthesize DBNs from constructs expressed in the Hap DSL. Scruff is Charles River's third-generation PPL (cra.com/projects/scruff/). It is an open-source library for the Julia programming language designed as a composable framework for efficient inference of complex generative systems (github.com/charles-river-analytics/Scruff.jl).

Concept of Operations

Putting the three parts together, iHap first ingests observations. It then constructs and samples a DBN of goal and mental-state constructs expressed in the Hap-based DSL. This sampling enables the iHap cognitive inverter to assess the likelihood that observations could have been generated by goal pathways. Once a cognitive inverter ingests the observations, synthesizes a DBN over the DSL, and infers probabilities teammate goals, a larger system, such as PSI-Coach, can act on the inference. PSI-Coach goes on to score likely causes of behavior and the inferred state of the mission, and then intervene to provide recommendations or other support. Figure 2 illustrates the function of the cognitive inverter in the larger PSI-Coach architecture.

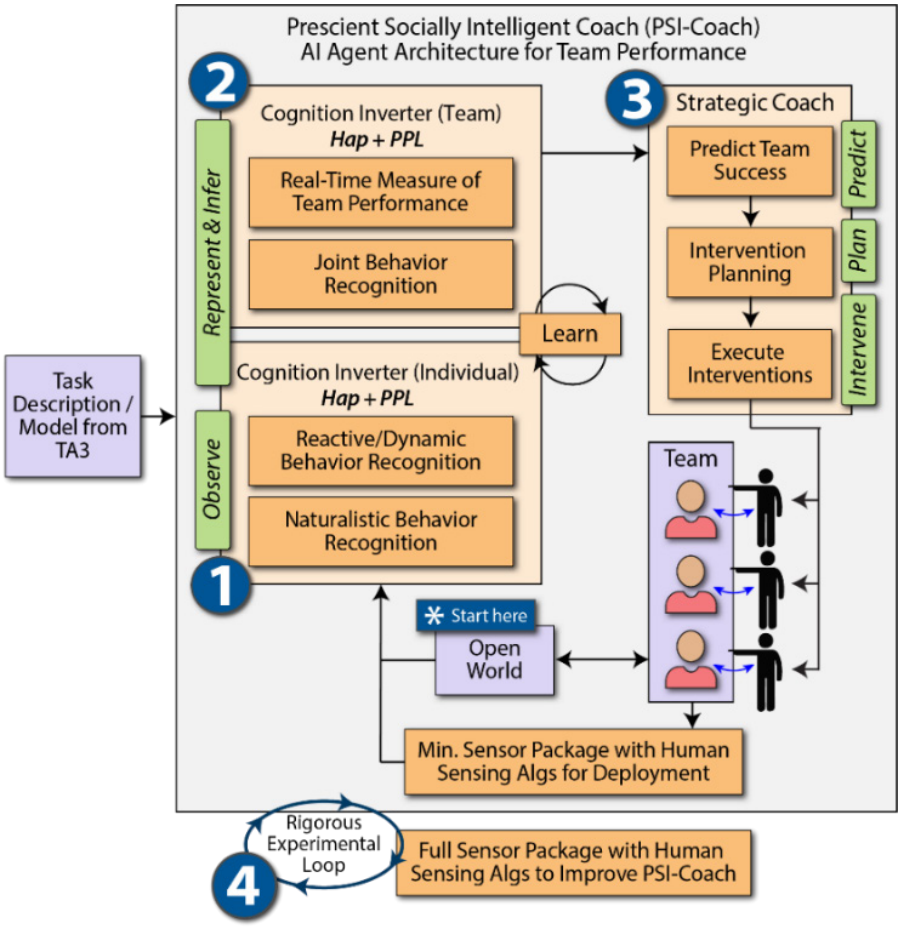

Figure 2. PSI-Coach functional architecture. *PSI-Coach uses a cognitive inverter to understand and act on team goals, mental states, and behaviors.*

In Figure 2, the process starts at the asterisk (*) with the cognitive inverter observing the team's interactions with the open world. It then (1) infers the goals, mental states, and behaviors of individual team members in detail—whether they are acting rationally or irrationally—and recognizes those goals, mental states, and behaviors in the presence of

the dynamic, and adaptive changes that people regularly exhibit in open worlds. Next, the cognitive inverter (2) infers the joint behaviors, shared mental models, and misaligned mental models of the team, and uses all of this inferred data to compute PSI-Coach's real-time measures of team performance. These measures are passed to the Strategic Coach (3), which predicts possible futures, assesses team effectiveness in those possible futures, plans potential interventions, and then executes interventions at the appropriate times to help the team achieve high performance. As PSI-Coach monitors team activities, repeating this cycle, the cognitive inverter is also continuously learning the strengths and weaknesses of the team and using that information to inform its predictions and intervention planning. PSI-Coach also includes (4) an offline, rigorous experimental cycle to inform research and improve the PSI-Coach system.

METHODS

Construction of the Minecraft testbed and human subjects data collection were carried out collaboratively among performers on the Defense Advanced research Projects Agency (DARPA) Artificial Social Intelligence for Successful Teams (ASIST) program. Colleagues at Aptima, Inc., led testbed construction and colleagues Arizona State University (ASU) led data collection. Development and testing of our cognitive inverter proceeded in three phases. In phase I, we collected initial data on human teams within the Minecraft-based testbed. In phase 2, we implemented a DBN for cognitive inversion without using a DSL and tested it during additional periods of human data collection. In phase 3, we implemented and tested the Hap DSL and full Inverse Hap cognitive inverter that uses probabilistic program synthesis to construct DBNs.

Human subjects research comprised three studies, following a research protocol approved by the ASU institutional review board (IRB) and DARPA's Human Research Protection Office (HRPO). Each study comprised approximately 20 experimental runs through the testbed. Experiment 1 tested individual participants on the SAR task, and experiments 2 and 3 tested teams of three participants. Participants played one of three roles: scouts moved quickly to locate casualties or move triaged casualties, medics triaged casualties and prepared them for extraction, and engineers removed rubble to clear paths/doorways and to free trapped teammates. Any role could search for and discover casualties. Once triaged, casualties could be moved to an extraction location matched to their triage category (A, B, or C).

After initial data collection, we annotated the data to capture domain knowledge. We identified strategies and goals, as well as hierarchical relationships between goals, subgoals, and behaviors (i.e., methods of achieving a goal via subgoals that eventually ground out in observable behavior). For example, the three roles had different goals, and individual participants might adopt different strategies for navigating the map based on their goal. Participants also sometimes became disoriented, unsure of their location and wandered until they reoriented themselves. Thus, a room or a casualty might be bypassed for any of several reasons. Some example reasons include the participant was carrying a victim, was going to help a teammate with a critical victim, was an engineer looking for blockages, was using a navigation strategy (e.g., "always turn left") that did not encompass the bypassed room, or was confused. The cognitive inverter's job is to infer the likelihoods of these types of mental states from observed behavior so that PSI-Coach could assess team performance and intervene when needed.

In phase 2, we constructed a DBN in Figaro over iterative development and testing against the phase I human data and encoded the domain knowledge developed from the annotations. The DBN is a probabilistic graph that traces paths from high-level goals or strategies through subgoals to observable behaviors. As a dynamic model, it steps through time via a state transition function representing the dynamics of the test environment to recognize active goals and subgoals capable of explaining the evolving behavioral record as new actions are observed. This phase 2 inverter did not perform probabilistic program synthesis over a DSL, but encoded features of Hap behavioral reactivity and variability in the hand-built linkages between belief nodes. After constructing the DBN, we integrated it into the larger PSI-Coach system and tested it in further human subjects research in the Minecraft testbed.

In phase 3 of development and testing, we are implementing and testing the Hap DSL and Inverse Hap cognitive inverter, iHap. Inverse Hap has two key features. First, it uses the same grammar as forward-running Hap, which enables the cognitive modeler (user) to quickly externalize their domain knowledge by encoding building blocks of agent goals and behaviors in the DSL. Second, it builds upon the inference capabilities of Scruff to convert the forward-running Hap constructs expressed in the DSL into a generative probabilistic model over possible agent states. A Hap agent's state at any point in time is represented in its Active Behavior Tree (ABT) and Working Memory. The current

set of goals and behaviors, and their current execution state, are represented in the ABT. Mental states are represented in Working Memory. Using the Scruff PPL, we invert Hap to recognize the likely ABT fragments and mental states that could have given rise to the sequence of actions that was observed. The ABT fragments, behaviors, and mental-state variables constitute the DSL—the "primitives" that can be probabilistically sampled and combined to construct hypotheses about the causes of the observed behavior.

The DSL provides preorganized goal and mental-state constructs expressed in the Hap cognitive modeling language. These Hap constructs capture domain knowledge, such as goals and relations between goals (causal paths from goals, to subgoals, to behaviors). The constructs also capture expressive complexities, such as that multiple goals can be active currently, some subgoals must be executed sequentially, and goals and behaviors may have context conditions, may succeed or fail, and may be achieved in more than one way. The expressivity of the constructs enables the cognitive inverter to explain nonlinearities in the observations (e.g., that actions resulting from multiple goals running in parallel may be interspersed in serial time, or that an initial attempt to achieve a goal by one method may fail part way through its subgoals, while a subsequent attempt, via different subgoals, may succeed). To create the Hap DSL, we ported key elements of the Hap architecture, grammar, and syntax from Java to Julia. We created Julia macros to abstract the complexities of the Julia code, which provides a compact way to write iHap programs and retains familiar Hap syntax.

Figure 3 depicts a high-level view of the iHap architecture. All DSL constructs (iHap macros embedded in Julia programming language) are interpreted and serialized into Julia source code for iHap programs (ABT definitions). An inference interpreter then takes the generated ABT definitions along with a definition of the agent's initial world state and synthesizes and runs a Scruff probabilistic model: the iHap cognitive inverter. The inverter is a DBN representation over possible ABTs and agent world states—hypothesized goal pathways, from top-level ultimate goals to observed actions. The DBN uses an asynchronous particle filter (aPF) as an inference algorithm. The aPF under the hood uses the expansion/contraction logic around the Hap engine as a generative model to create probability distributions over possible ABTs. We can then assert evidence against this probability distribution by observing actions that a human performs to then infer the most likely ABT that would have generated said action(s). The inverter can thus be queried for likelihoods and posteriors at each node of the DBN.

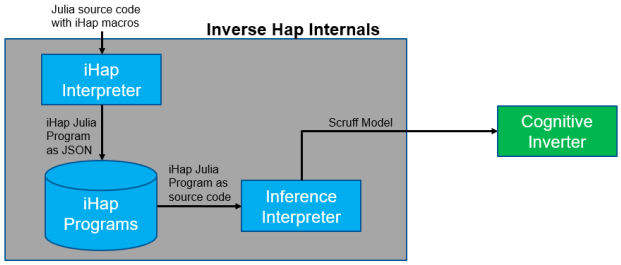

Figure 3: Inverse Hap cognitive inverter architecture.

iHap extends the forward Hap system by (1) replacing random choices in Hap with probability distributions inferred through the PPL; (2) using the Working Memory variables as latent variables to be inferred; (3) adding latent internal variables for goal switching and reactive changes, so additional dynamic changes can be learned and recognized; and (4) adding dynamic noise to deterministic choices in the behavior models (e.g., step skipping and changes in step ordering, to support learnable exceptions to these choices). With these extensions, when the cognitive inverter sees a sequence of actions that does not fit a single goal or activity, but instead fits a mix of partially executed goals, it can recognize behaviorally plausible combinations. By only recognizing behaviorally plausible combinations, the inference is efficient and avoids spurious recognitions.

iHap creates a Hap ABT of possible paths to observations given the domain knowledge, then uses particle filtering inference to determine the posterior belief of each node. Overall, given that one has modeled a set of behaviors, iHap ingests observations, generates Hap agent fragments that can produce them as nodes in a DBN, and assesses the likelihood that each node had a role in causing the observations.

RESULTS

Figaro DBN

We ran experiments with the Figaro-encoded cognitive inverter using observations of human team behavior collected in the Minecraft-based SAR task. The results demonstrated that a DBN that encoded goal-oriented SAR domain

knowledge and important elements of human strategic and behavioral variability could recognize team goals from behavioral observations. Figure 4 and Figure 5 illustrate cognitive inverter results and their use by the larger PSI-Coach system overlayed on testbed maps.

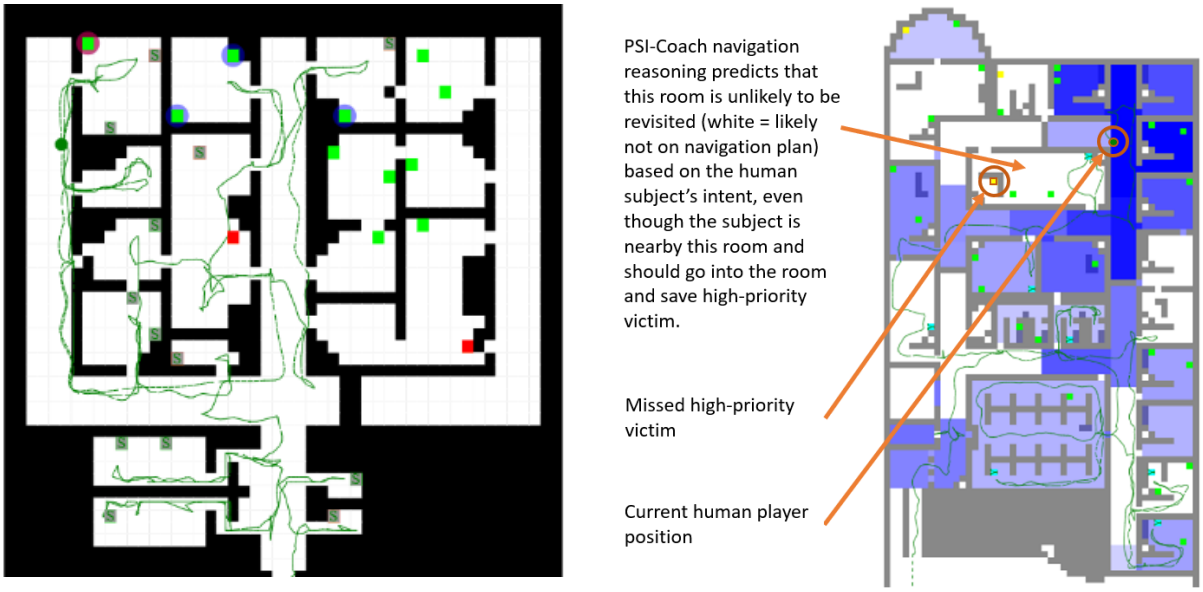

Figure 4. Inference of triage strategy (left) and prediction of future search behavior (right), illustrated in top-down views of building interiors, showing a participants' navigation paths (green lines) and casualty locations (colored squares).

In Figure 4, the left-hand panel illustrates the cognitive inverter infereence of the triage strategy from observed navigation behavior. Casualties denoted by a circular blue highlight around a green square were likely bypassed on purpose. Casualties denoted by a circular red highlight around a green square were likely bypassed inadvertently. In the right-hand panel, PSI-Coach predicts future search behavior from inferred triage strategy. Darker blue denotes areas likely to be (re)visited. Unshaded areas are unlikely to be (re)visited. PSI-Coach can then intervene to prompt a teammate to check for missed high-priority casualties that are in areas unlikely to be visited.

In Figure 5, a participant in the medic role has been ignoring team communication prior to and within the area of the blue box, entering rooms that have been communicated as clear. PSI-Coach waits to issue an intervention at the start of the orange box, when the cognitive inverter infers that the medic is about to ignore team communication again. After the intervention, the medic turns around, avoiding a repeated mistake; starts reliably following better team process; and uses team communication more consistently.

Overall, cognitive inversion automatically identified team process problems unique to different teams and their situation dynamics. Based on such inferences, PSI-Coach provided timely, tailored intervention content that improved team processes within 60 seconds. Cognitive inversion showed a 35% increase over rule-based comparison systems' real-time inferences ($p<0.05$) and led PSI-Coach to exhibit a 42%–68% increase in agreement with human coaches on interventions over a baseline inference method ($p<0.05$).

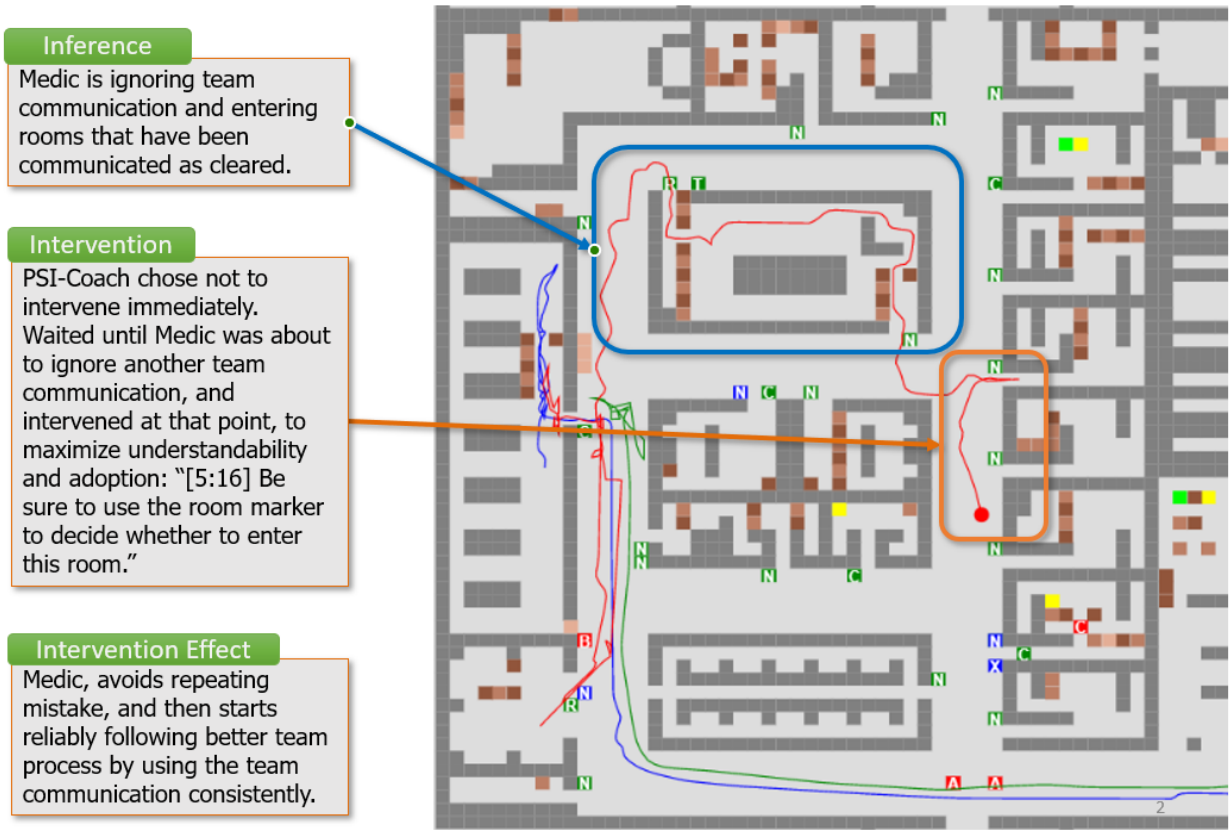

Figure 5. Demonstration of medic decision-making before and after intervention from PSI-Coach (the red line traces the path of a participant in a medic role, with the current location shown as red to in middle-left of the figure).

Inverse Hap

We implemented the PPL version of the Hap language with full-scope features, including shared memory and language scoping and post hoc probabilistic modeling of agent mental activity. A PPL factor construct allows approximate Bayesian scoring of sequences of agent behavior. iHap allows aPF inference with soft sampling controlled by factor. In general, inference in a full-scope system presents scalability challenges: agent programs express probability distributions over high-dimensional combinatorial spaces, with constraints expressed through approximate Bayesian scoring. Creating a DSL from a cognitive modeling architecture such as Hap addresses these constraints, yielding a novel perspective on inference over agents: parsing sequences of agent action observations as generated from a probabilistic grammar. iHap synthesizes agent programs from agent grammars applied to behavioral subsequences. Our expanded factor-based inference construct supports island-driven inference, performing inference over behavior fragments and combinations of fragments for increased scalability and robustness. This extends the inference capabilities from traditional inference over random choices facilitated by human cognitive modeler to inference over entire behavior models from observed actions in the domain. In addition, our designs for inference compilation technology speed up inference and increase model coverage.

Presently, we have created simple tests for the iHap cognitive inverter and run small experiments to verify component functionality and integration. We have not tested iHap on the human data or integrated it into the larger PSI-Coach architecture. To test iHap, we implemented simple iHap models (an example is illustrated in Figure 6), which nonetheless exhibit varying possible behaviors and actions to explain observations. In the example in Figure 6, the observation is that a searcher has walked past a room without checking it for casualties. This may have been accidental (an event with increased likelihood if the searcher is a novice) or it may have been strategic (e.g., because the searcher was going to help a medic with a critical casualty).

For the example in Figure 6, we ran three tests, wherein we passed the resultant Scruff model an observation that the searcher missed a room and was either novice, expert, or of unspecified skill. The model successfully returned probabilities of the missed versus skipped explanations based on evidence posted about searcher skill and passing by a room without checking. These types of tests, albeit simple, indicated that the iHap pipeline functions as intended, from observations, through probabilistic program synthesis over DSL constructs, to causal inference, culminating in posterior probabilities at each node.

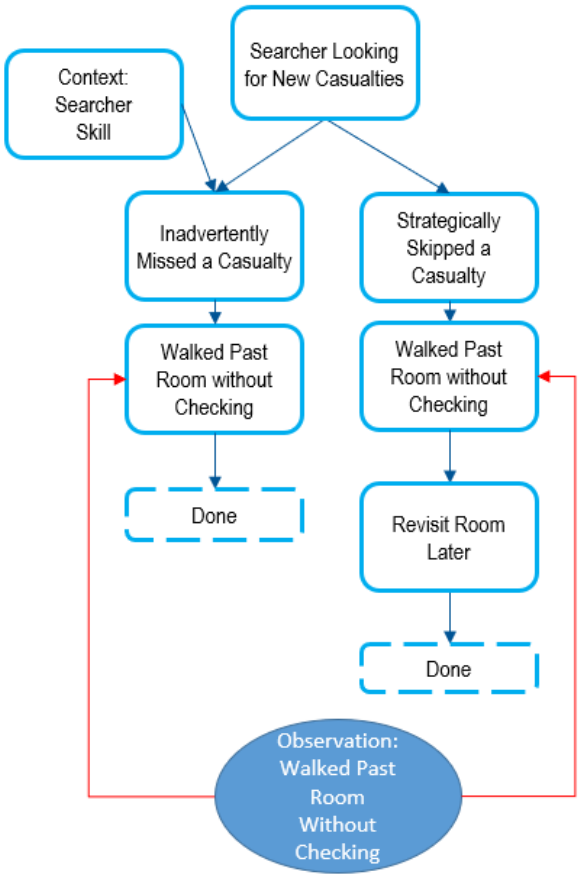

Figure 6: Goals, mental states, and an observation for a simple iHap proof-of-concept test.

CONCLUSION

People are complex and dynamic; they do multiple things at the same time, switch between goals and tasks, pause activities and return to them later, and react to opportunities and changes in the world. Traditional plan recognition and activity recognition systems struggle to handle this dynamic behavior. Plan recognition systems recognize a user's specific stage in a rigid plan but fail in the complexity of an open world. Activity recognition systems (e.g., those using deep learning, rule-based systems, or Bayesian graphical models) robustly recognize coarse-grained activities within the dynamic complexity, but they fail to provide the specific context of where in a plan an individual is stuck. Cognitive inversion provides detailed recognition of team members' goals and the behavior execution state in these natural, overlapping flows of human activity by inverting a cognitive model's mechanisms for such reactive, adaptive, and dynamic cognitive processes. Our implementation of cognitive inversion enables PSI-Coach to monitor team members and intervene during training or operations to improve team performance.

Our results show that cognitive architectures combined with probabilistic programming to implement cognitive inversion can be used to represent important human mental model variations, including intent, task strategy, task execution, and resource usage and knowledge, among other factors. Cognitive inverter algorithms can infer dynamically changing mental models (e.g., unexpected strategies, changes in intent) from observations of behavior and can predict future execution of strategies to enable well-timed intervention points, similar to human coaches.

We implemented two cognitive inverters. The first cognitive inverter encoded domain knowledge and behavioral variability directly into a DNB in the Figaro PPL. We used this version to test cognitive inversion and PSI-Coach with human data prior to developing a DSL from a cognitive architecture. While this approach is successful, it requires the user to have diverse technical expertise. The user needs to know the principles of probabilistic programing and DBNs, the specific PPL (e.g., Figaro, a Scala library), as well as the domain knowledge (e.g., strategies and pitfalls of performing SAR). The second cognitive inverter used a DSL based on a cognitive architecture, Hap. The use of macros to create goal, strategy, and behavioral constructs in the cognitive architecture means the user only needs to how to build cognitive models (e.g., in Hap) based on domain knowledge. The use does not need to know Julia, Scruff probabilistic modeling, or how to create a DBN; the DSL-based cognitive inverter does the work of synthesizing and running the inverted cognitive model.

Cognitive inversion can benefit training because an inverter-equipped intelligent training system can know the ground-truth state of the training range and scenario and use that knowledge to intervene at opportune moments. For example, when intervening on a searcher who accidently missed a casualty earlier, PSI-Coach could wait until the searcher was

near the casualty's location again to prompt the searcher about the earlier mistake. Cognitive inversion can benefit operations because an inverter-equipped intelligent decision aid can analyze incoming data to infer adversarial tactics, techniques, and procedures (TTPs), suggest and provide confidence estimates over alternative hypotheses about adversary TTPs and strategies/goals (PSI-Coach's cognitive inversion step), and suggest complementary courses of action (PSI-Coach's intervention step). Potentially, the cognitive inverter can mitigate confirmation bias and other cognitive biases or point an analyst to where observations are missing that, if discovered, could resolve uncertainty.

ACKNOWLEDGMENTS

This work was performed as part of the Defense Advanced Research Projects Agency (DARPA) Information Innovation Office (I2O) Artificial Social Intelligence for Successful Teams (ASIST) program, under Contract HR0011-19-C-0126. The authors thank Joshua Elliot, Brian Sandberg, Tanya Tanner, and Jennifer Mack for their support and direction on this project. We would also like to thank our extended project team, who were essential to making PSI-Coach a successful effort. We are grateful to colleagues at Carnegie Mellon University for discussions of theories of intelligence and for contributions to experimental design, colleagues at Arizona State University for managing the human subjects research, and colleagues at Aptima for construction of the Minecraft™ testbed.

REFERENCES

Arora, N. S., Russell, S., and Sudderth, E. (2013). NET-VISA: Network processing vertically integrated seismic analysis. *Bulletin of the Seismological Society of America*, *103*(2A), 709–729. https://doi.org/10.1785/0120120107

Bates, C., Battaglia, P., Yildirim, I., and Tenenbaum, J. B. (2015). Humans predict liquid dynamics using probabilistic simulation. *CogSci*. https://cogsci.mindmodeling.org/2015/papers/0040/paper0040.pdf

Kulkarni, T. D., Kohli, P., Tenenbaum, J. B., and Mansinghka, V. (2015). Picture: A probabilistic programming language for scene perception. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4390–4399. https://doi.org/10.1109/CVPR.2015.7299068

Loyall, A. B., Reilly, W. S. N., Bates, J., & Weyhrauch, P. (2004). System for authoring highly interactive, personality-rich interactive characters. *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 59–68. https://doi.org/10.1145/1028523.1028532

Loyall, B. (1997). Believable agents: Building interactive personalities. Carnegie Mellon University. https://www.cs.cmu.edu/afs/cs/project/oz/web/papers/CMU-CS-97-123.pdf

Pfeffer, A. (2016). *Practical Probabilistic Programming* (Version 1st, 1st ed.) [Computer software]. Manning Publications Co.

Pfeffer, A., Harradon, M., Campolongo, J., and Cvijic, S. (2021). Unifying AI algorithms with probabilistic programming using implicitly defined representations. arXiv preprint. https://doi.org/10.48550/arXiv.2110.02325

Sliva, A. L., Gorman, J., Voshell, M., Tittle, J., & Bowman, C. (2016). Combining cognitive engineering and information fusion architectures to build effective joint systems. *SPIE Defense + Security*, 98310M. https://doi.org/10.1117/12.2223926