AIIDE 2012 Doctoral Consortium
AAAI Technical Report WS-12-18

Assistant Agents For Sequential Planning Problems

Owen Macindoe
Advisors: **Leslie Pack Kaelbling** and **Tomas Lozano-Perez**
{owenm, lpk, tlp}@mit.edu
Computer Science and Artificial Intelligence Laboratory
MIT, Cambridge, Massachusetts, 02139

Abstract

The problem of optimal planning under uncertainty in collaborative multi-agent domains is known to be deeply intractable but still demands a solution. This thesis will explore principled approximation methods that yield tractable approaches to planning for AI assistants, which allow them to understand the intentions of humans and help them achieve their goals. AI assistants are ubiquitous in video games, making them attractive domains for applying these planning techniques. However, games are also challenging domains, typically having very large state spaces and long planning horizons. The approaches in this thesis will leverage recent advances in Monte-Carlo search, approximation of stochastic dynamics by deterministic dynamics, and hierarchical action representation, to handle domains that are too complex for existing state of the art planners. These planning techniques will be demonstrated across a range of video game domains.

Introduction

There are many domains for which it would be useful to have an AI assistant, from helping around the house to managing your work. Video games in particular offer a natural domain for AI assistants, often featuring non-player characters (NPCs) that help human players achieve their goals in the game. Since human intentions are not directly observable and are a crucial part of the domain in a collaborative game, AI assistants ought ideally to act with this partial observability in mind, attempting to simultaneously understand the unobservable intentions of their human counterparts and to act to help bring them about. This research explores a variety of approaches to automatic planning for AI assistants that take this partial observability into account.

In recent years planning-based approaches to managing AI behavior have been successfully applied in an increasing number of commercial video games, including goal-oriented action planning in the F.E.A.R. series (Orkin 2004) and hierarchical task network planning in the Killzone series (Straatman, Verweij, and Champandard 2009). In parallel to this work, researchers have been interested in inferring human goals within a game environment from the actions that they take (Ha et al. 2011).

This thesis will present approaches to planning in video games that empower AI assistant characters to combine goal inference with action planning. In particular we will explore approaches that plan in belief space, which is the space of probability distributions over world states that represent the assistants' uncertainty about the current state of the world. Planning in belief space allows AI assistants to account for their uncertainty about their human counterpart's intentions, and to plan actions that help to disambiguate them.

As a motivating example, consider the simple collaborative game Cops and Robbers, shown in Figure 1. In this game the player and their AI assistant are cops, chasing robbers who are fleeing from them through the corridors of a building full of twisting passages and one-way doors. The object of the game is to catch any one of the robbers as quickly as possible, but it takes two cops to overpower a robber, so the player and their sidekick must coordinate to corner a robber and catch them. The presence of one-way doors means players have to be careful about the routes they take or they could find themselves wasting a lot of time. The core challenge for planning as an AI assistant in Cops and Robbers is to infer which robber the human intends for the two of you to catch and to plan moves in order to cut off their escape.

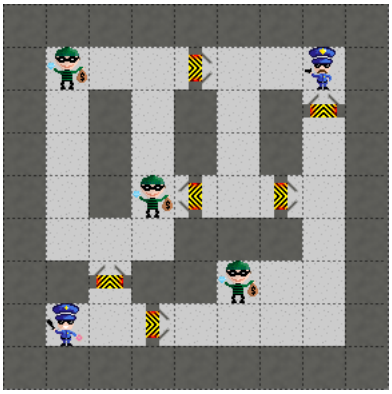

Figure 1: The Cops and Robbers play field. The human is in the top right, the assistant is in the bottom left. Chevrons indicate the direction of one-way doors.

Formulating the AI assistant problem

We will call the problem of simultaneously inferring and supporting human intentions the AI assistant problem. The problem is both partially observable and stochastic, even for games which are themselves fully observable and deterministic. This is because the presence of a human actor in the game adds their intentions as an unobservable state variable and also adds stochasticity, since human actions typically can not be perfectly predicted.

The choice of problem formulation has a strong influence on tractability of the planning problem. One possibility is to place no constraints on the kinds of expected human behavior, in which case the problem is a partially observable stochastic game, for which optimal planning is NEXP^{NP}-hard (Goldsmith and Mundhenk 2008). A less extreme alternative is to assume that the humans will act as if they had previously conferred with the assistant and come up with an optimal joint plan that respects the uncertainty of the domain, in which case the problem is a decentralized partially observable Markov decision process (Dec-POMDP), for which optimal planning is still NEXP-complete (Bernstein et al. 2002).

In this thesis work we assume that the assistant has some set of models of human behavior, one for each possible human intention, that allow it to predict what a human would do given the current state of the world. This allows us to formulate the problem as a partially observable Markov decision process (POMDP), for which finding an optimal plan is still PSPACE-complete in the finite horizon case and undecidable in the infinite horizon case (Papadimitiou and Tsitsiklis 1987; Madani, Hanks, and Condon 1999), but there are a variety of approximate solution methods that could scale to a video game domain.

Informally, a POMDP specifies the reward structure of the game, actions available to the assistant, the states of the game including unobservable state variables such as human intentions and other hidden variables, the dynamics of the game that govern how the state variables change as the assistant takes actions, and the observation dynamics that govern what the assistant perceives in the current state.

Because state variables are not directly observable in a POMDP, an AI assistant must maintain a probability distribution over the possible states of the world in which it could be, given its history of action and observations. This is called a *belief*. Planning in POMDPs involves finding a function from beliefs to actions called a *policy*. An important related function is the *value function* of a policy, which maps a belief to the the expected reward of acting according to the policy, starting from that belief.

Human models form a key part of the dynamics of the POMDP formulation, with the true human model and any of its internal states being hidden state variables in the POMDP. Figure 2 shows an example of how a POMDP can be constructed from the dynamics of a fully observable game and a set of human models. Given a formulation of the game as fully observable multi-agent MDP (MAMDP), we can generate the state and observation dynamics for a POMDP representing the AI assistant's perspective by connecting the human model to the state update dynamics. Figure 2 demonstrates the flow of a state update in the resulting POMDP. After receiving an action from the assistant, the output of the MAMDP's state update is passed to the input of the unobserved human model, producing a human action. This action is then passed the back to the MAMDP again for another state update, finally returning the human action along with the game state as an observation for the assistant.

Variations on this structure include embedding a multi-agent POMDP, rather than an MAMDP, in which case the human model would be passed observations, rather than states, and not including human actions or any state variables as part of the assistant's observation.

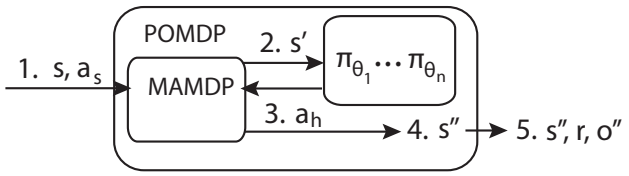

Figure 2: Example of a POMDP formed from a MAMDP and a set of human models. The dynamics of a state update are: 1. The simulator is passed a state s and assistant action a_s. 2. The MAMDP dynamics update the state to s'. 3. The human model π_θ for human type $\boldsymbol{\theta}$ in s' selects the human action $a_h = \pi_\theta(s')$. 4. The MAMDP dynamics update the state to s''. 5. The POMDP observation dynamics and reward signal produce o'' and r, returning them with the successor state s''.

Planning approach

Ideally, given a POMDP description of a game, one would like to find the policy that maximizes the assistant's expected future rewards, known as the optimal policy. Indeed, methods for computing optimal policies exist, such as the witness algorithm, which is part of a family of dynamic programming algorithms called value iteration (Kaelbling, Littman, and Cassandra 1995). However, because of the known PSPACE-completeness of optimal planning in POMDPs, we cannot guarantee that an optimal policy can be tractably found. Instead we will attempt to approximate the optimal policy.

A key source of intractability in value iteration approaches is that they often attempt represent the value function explicitly, which involves finding a complete mapping from an intractably large belief space to the space of expected rewards. We present three alternatives that avoid explicitly computing the belief space value function: collaborative Q_{MDP}, partially observable Monte-Carlo cooperative planning (POMCoP), and maximum likelihood A* (MLA*).

Collaborative Q_{MDP}, is based on based on the Q_{MDP} approximation, which uses a value function derived from offline planning in state space, weighted by its current belief, to select actions (Littman, Cassandra, and Kaelbling 1995). This corresponds to acting under an optimistic assumption that after its next action the world will become fully observable to the AI assistant.

POMCoP uses the POMCP algorithm for POMDP planning (Silver and Veness 2010). POMCP plans online us-

ing UCT search (Kocsis and Szepsvari 2006) in the tree of action and observation sequences to approximate the quality of each possible action, conditioned on its current belief, and performs Monte-Carlo belief updates after receiving observations using a particle filter. This tree search approach avoids having to consider the entire space of beliefs, focusing instead on belief states that are reachable from the current belief and on actions that are potentially beneficial as judged by Monte-Carlo approximation.

MLA* applies online A* search to a determinization of belief space derived by assuming that any action taken by the assistant will result in the most likely observation according to the POMDP's state and observation dynamics. States in the search are still belief states, computed using a belief update derived from the POMDP's true state and observation dynamics. The heuristic and goal criteria for the search are game dependent. If in the course of following the A* plan the assistant does not receive the most likely observation, the assistant replans from the current belief.

Human models

Recall that to formulate the AI assistant problem as an POMDP we made the assumption that we have a set of models of human behavior that specify how a human would act in the game given some particular intention. A human model is itself a stochastic policy, i.e. a stochastic mapping of game states to actions. The problem of generating such a set of models is a challenge in itself.

One possibility is to consider a set of possible goals that the human may be trying to achieve and then compute the optimal joint policy for achieving those goals, assuming that the human and the AI assistant could share beliefs. This amounts to the human acting as if they had perfect communication with the assistant and were able to tell them exactly what to do. We could then use the human's half of the joint policy as a human model. We will call this the joint planning model.

This approach is particularly attractive when the human's intent is the only hidden state variable, since it allows us to use state space planning to find a policy for each possible human intent, which is much easier than finding a joint plan in belief space. This approach has notably been taken in the CAPIR planning framework (Ngyuen et al. 2011). When this is not the case however, finding this joint policy involves optimal planning in a POMDP, requiring approximation techniques as usual.

Another possibility is to leverage domain knowledge about the game in order to construct a set of heuristic human models that represent intuitively plausible human behaviors. We will call this the heuristic model.

If we have a set of observational data from humans playing the game, we can use machine learning to infer a policy and use it as a human model. This modeling approach has been demonstrated by researchers for a variety of machine learning techniques including decision trees (Barrett, Stone, and Kraus 2011) and reinforcement learning (Tastan and Sukthankar 2012). We will call this general approach the machine learning model.

Progress

We have implemented and tested a collaborative Q_{MPD} planner for two collaborative pursuit games. One game is the Cops and Robbers game mentioned previously and the other is a similar game, but without one way doors and with stochastic movement from the enemies rather than fleeing behaviors, called the Ghost Game. Both games also featured communication actions that let the assistant ask the human about their intentions and let the human respond.

Since each of these games was fully observable, aside from the human's goal which in each case we took to be the particular enemy the human was pursuing, we were able to use the joint planning model outlined in the previous section. We formulated each game as a MAMDP. We then formed one human model per possible target enemy by using value iteration in state space to solve for a joint policy for human and assistant actions. Using the human halves of the joint policies as models, with a small amount of noise introduced, we formulated the POMDP as shown in Figure 2. Finally we again solved for the optimal policies for the AI assistant interacting with each of the human models, giving us the state space policies required for Q_{MPD} planning.

To update beliefs for the planner we used the value function computed during the construction of the human models and made the assumption that humans choose actions noisily, but in proportion to the expected value of their outcomes, following MDP modeling work from cognitive science (Baker, Saxe, and Tenenbaum 2009).

The key weakness of QMDP planning has been that the time and space cost of value iteration is dependent on the size of the game's state space, which is typically exponential in the number of state variables. Additionally, plans made in state space will never include information gathering actions, in particular communication actions. We experimented with adding local search and computing action entropy (Cassandra, Kaelbling, and Kurien 1996) to add information gathering behaviors, but with mixed results. These concerns led us to focus more strongly on belief space search approaches.

We have implemented a POMCoP planner for the Ghost Game, Cops and Robbers, and a collaborative search game called Hidden Gold, in which the human and the assistant must search the environment for treasure and dig it up together. Hidden Gold also included a shout action, which did nothing other than create a yell observation.

For each game we used heuristic human models since we wanted to work with state spaces larger than a joint policy could be computed for. For the two pursuit games the heuristic model moved myopically via noisy A* to its target enemy. For Hidden Gold the human model moved via noisy A* to a random hiding spot that it hadn't yet visited and occasionally yelled when it had found the treasure.

POMCoP performed on par with collaborative Q_{MDP} on the Ghost Game and outperformed it on Cops and Robbers, where belief space planning allowed it to both take communication actions more effectively and also take stalling actions that hedged against chasing the wrong robbers and becoming slowed down by the maze of one-way doors. POMCoP's plan quality varied with the amount of search time it was allocated, since more time allowed it to improve its

Monte-Carlo estimates of the optimal value function. Given an average of 20 seconds of search per move, POMCoP could perform well on maps 4 times the size of the maximum that collaborative Q_{MDP} could handle. When restricted to 2 seconds of search, the size of maps playable was only marginally larger. Our analysis suggested that the branching factor of the belief space search and long search horizons were responsible for these performance problems.

Work remaining

We are also currently working on implementing MLA* planning, which we hope will be able to construct longer plans than our current POMCoP planner by cutting reducing branching factor. We are also considering modifying POMCoP to use the same determinized dynamics.

To help further increase the planning horizons in POMCoP and also to improve the efficiency of MLA* we intend to introduce macro actions, which compress action sequences into a single action for planning purposes (Lim, Hsu, and Sun 2011). We expect macro actions to be particularly helpful for movement, since in our current planners a great deal of the maximum reachable search horizon is wasted on straight-forward navigation.

The motivation behind pushing for a tractable solution to the AI assistant planning problem is to produce a planner that can actually be applied to complex games. Beyond our current set of games we intend to apply our planners to Dearth, a video game developed as part of a collaboration between CSAILs Learning and Intelligent Systems group and the Singapore-MIT GAMBIT Game Lab, which was specifically designed to demonstrate the potential of MDP planning paradigms in developing AI assistants for games. Dearth already includes a limited collaborative Q_{MDP} planner which was the inspiration for our current planner. We also intend to apply these techniques to Ghostbusters, a game developed at NUS in tandem with the CAPIR planning framework (Ngyuen et al. 2011), an open source commercial-scale video game, and a smaller scale game that we will develop in collaboration with GAMBIT UROPs.

Much work remains to be done on acquiring human models. We will start by implementing a machine learning model following previous work modeling human policies as decision trees (Barrett, Stone, and Kraus 2011). We are also investigating the possibility of extending POMCoP to automatically generate human models by assuming that the human acts optimally with respect to the AI assistant's belief about their intentions and selecting actions for them accordingly during Monte-Carlo simulations.

Although our informal qualitative tests of our AI assistants alongside humans have so far been positive, we have only empirically compared the our planning systems against each other when paired with simulated human partners. We intend to run user studies comparing the performance of the different planners with real human partners and collecting human judgements of AI assistant quality.

Although optimal planning is not possible for the scale of games in which we are interested, we intend to empirically compare the quality of the plans produced by our planners to off-the-shelve POMDP solvers for small domains. This will be the first step in proving bounds on the quality of the plans produced by our systems.

We will also perform an analysis of the completeness of MLA* planning, which we expect to be a challenge, since we expect the determinization process to make parts of belief space unreachable by the search.

References

Baker, C. L.; Saxe, R.; and Tenenbaum, J. B. 2009. Action understanding as inverse planning. In *NIPS*.

Barrett, S.; Stone, P.; and Kraus, S. 2011. Empirical evaluation of ad hoc teamwork in the pursuit domain. In *AAMAS*.

Bernstein, D. S.; Givan, R.; Immerman, N.; and Zilberstein., S. 2002. The complexity of decentralized control of markov decision processes. *Mathematics of Operations Research* 27(4).

Cassandra, A. R.; Kaelbling, L. P.; and Kurien, J. A. 1996. Acting under uncertainty: Discrete Bayesian models for mobile robot navigation. In *IEEE/RSJ Intelligent Robots and Systems*.

Goldsmith, J., and Mundhenk, M. 2008. Competition adds complexity. *NIPS* 20.

Ha, E. Y.; Rowe, J. P.; Mott, B. W.; and Lester, J. C. 2011. Goal recognition with markov logic networks for player-adaptive games. In *AIIDE*.

Kaelbling, L.; Littman, M.; and Cassandra, A. 1995. Planning and acting in partially observable stochastic domains. *AI* 101.

Kocsis, L., and Szepsvari, C. 2006. Bandit based monte-carlo planning. In *NIPS*.

Lim, Z. W.; Hsu, D.; and Sun, L. 2011. Monte carlo value iteration with macro actions. In *NIPS*.

Littman, M. L.; Cassandra, A. R.; and Kaelbling, L. P. 1995. Learning policies for partially observable environments: Scaling up. In *ICML*.

Madani, O.; Hanks, S.; and Condon, A. 1999. On the undecidability of probabilistic planning and infinite- horizon partially observable decision problems. In *AI*.

Ngyuen, T. H. D.; Hsu, D.; Lee, W. S.; Leong, T. Y.; Kaelbling, L. P.; Lozano-Perez, T.; and Grant, A. H. 2011. Capir: Colaborative action planning with intention recognition. In *AIIDE*.

Orkin, J. 2004. Symbolic representation of game world state: Towards real-time planning in games. In *AAAI Workshop on Challenges in Game AI*.

Papadimitiou, C. H., and Tsitsiklis, J. N. 1987. The complexity of Markov decision processes. *MoOR* 12(3).

Silver, D., and Veness, J. 2010. Monte-Carlo planning in large POMDPs. In *NIPS*.

Straatman, R.; Verweij, T.; and Champandard, A. 2009. Killzone 2 multiplayer bots. In *Paris Game AI Conference*.

Tastan, B., and Sukthankar, G. 2012. Learning policies for first person shooter games using inverse reinforcement learning. In *AIIDE*.