The AAAI-17 Workshop on
Human-Machine Collaborative Learning
WS-17-11

Active Preference Elicitation for Planning

Mayukh Das
School of Informatics & Computing
Indiana University, Bloomington, IN

Md. Rakibul Islam
EECS
Washington State University
Pullman, WA

Janardhan Rao (Jana) Doppa
EECS
Washington State University
Pullman, WA

Dan Roth
Department of Computer Science
University of Illinois at Urbana-Champaign, Urbana, IL

Sriraam Natarajan
School of Informatics & Computing
Indiana University, Bloomington, IN

Abstract

We consider the problem of actively eliciting preferences from a human by a planning system. While prior work in planning have explored the use of domain knowledge and preferences, they assume that the knowledge must be provided before the planner starts the planning process. Our work is in building more collaborative systems where a system can solicit advice as needed. We verify empirically that this approach lead to faster and better solutions, while reducing the burden on the human expert.

Introduction

Planning under uncertainty has been augmented with human input in several different directions (Tan and Pearl 1994; Dean et al. 1995; Boutilier, Dean, and Hanks 1995; Myers 1996; Huang et al. 1999; Allen and Ferguson 2002; Brafman and Chernyavsky 2005; Sohrabi and McIlraith 2008). One key research thrust in this direction is that of specifying preferences to the planner in order to reduce the search over the space of plans. While successful, most of the preference specification approaches required that the human input be provided in advance before planning commences.

We propose a framework in which the planner actively solicits preferences as needed. More specifically, our proposed planner computes the uncertainty in the plan explicitly and then queries the human expert for advice based on this uncertainty as needed. This approach not only removes the burden on the human expert to provide all the advice/suggestions/preferences upfront but also allows the planner to focus on the most uncertain regions of the plan space and query accordingly. Thus it avoids the humans from providing advice about trivial/most obvious regions of the plan space and instead focus on the harder part of the search from the planner's perspective.

In this work-in-progress paper, we present our initial algorithm for active preference elicitation in planning. As far as we are aware, this is the first work in this direction for planning. We consider the hierarchical task network planner for this task as it allows for seamless interaction with the humans who solve problems by decomposing them into smaller problems. We evaluate our algorithm on free cell domain where we compare against several baselines. Our initial results show that this collaborative approach allows for more efficient and effective problem solving compared to the standard approaches and to the approach of providing all the inputs in advance.

Background and related work

Preference elicitation (Boutilier et al. 1997; 2004; Myers 1996; Huang et al. 1999; Brafman and Chernyavsky 2005; Sohrabi and McIlraith 2008) has been explored inside automated planning, in reinforcement learning and inverse reinforcement learning (Maclin and Shavlik 1994; 1996; Torrey et al. 2005; Natarajan and Tadepalli 2005; Judah et al. 2010; Kunapuli et al. 2013). Most of the above approaches, though effective, require upfront encoding of knowledge/preference/advice. Recently, Odom et al.'s work on active advice seeking in IRL (Odom and Natarajan 2016), presented a framework to actively query for advice from human on uncertain states.
Preference based HTN planning (Sohrabi, Baier, and McIlraith 2009) relate closely to the HTN based framework we adopt and how the decision making process utilizes the preferences, but differ on how such preferences are acquired and encoded. Our work, is sort of an initial step towards a robust human-agent collaborative planner as conceptualized by Allen et. al.(2002).

Active preference elicitation

Previous approaches to active advice seeking have been explored in the context of inverse RL where a set of demonstrations are provided along with access to the human expert. The agent then learns the reward function from the combination of trajectories and advice. In a planning problems, however, there are no demonstrations available from which we can learn policies and get policy level uncertainty. Hence, for uncertainty based querying in planning, we must go beyond learning from demonstrations. As the goal here is not of recovering the underlying reward function, human suggestion/feedback acts as a way to prune the search space (see Figure 1). This is in contrast to Odom et al.'s active advice seeking, where the effect of the advice is included as an induced bias in the optimization function.

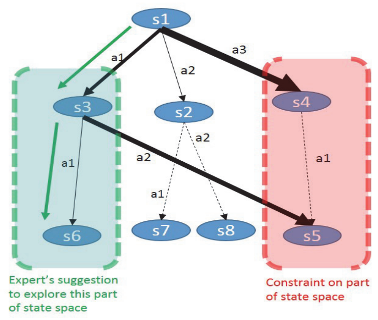
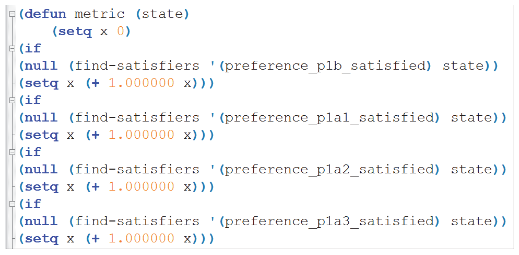

s1
a1
a2
a3
s3
s2
s4
a1
a2
a2
a1
a1
s6
s7
s8
s5
Expert's suggestion to explore this part of state space
Constraint on part of state space

Figure 1: Decision making as search space pruning

Preference on HTN planning

We used a Hierarchical Task Network (HTN), as the underlying planner (Ghallab, Nau, and Traverso 2004). It is a forward search planner which searches in the space of tasks and subtasks. It proceeds by decomposing a task, in its current list of tasks, into possible subtasks based on the *methods* defined in its domain knowledge. If a task is primitive (solvable by an atomic action) it is removed from the list of tasks, the action applied to the state and is added as a step in the current partial plan. Other *non-primitive* tasks are then decomposed further. It implicitly allows for several levels of abstraction, which would otherwise have be achieved via state clustering, since tasks are nothing but an abstraction for a sub-sequence of actions. Hence, preferences in an HTN planner results in pruning of the task decomposition tree. Hierarchical decomposition is closer to human reasoning and thus allows for seamless interactions.

Sohrabi et al., (2009) pruned the task decomposition tree by scoring the quality of decomposition using the hard-coded metric function, which just measures how well the task decompositions adhere to the preferences (Figure 2). Our approach evaluates the quality of the decomposition not just based on adherence to preferences but optimality of the partial plan as well.



```
(defun metric (state)
    (setq x 0)
(if
(null (find-satisfiers '(preference_p1b_satisfied) state))
(setq x (+ 1.000000 x)))
(if
(null (find-satisfiers '(preference_p1a1_satisfied) state))
(setq x (+ 1.000000 x)))
(if
(null (find-satisfiers '(preference_p1a2_satisfied) state))
(setq x (+ 1.000000 x)))
(if
(null (find-satisfiers '(preference_p1a3_satisfied) state))
(setq x (+ 1.000000 x)))
```

Figure 2: Hard coded metric in problem definition

Uncertainty in HTN

Uncertainty in HTN planning can be expressed via a distribution over *methods* (i.e. over the alternatives of decomposing a task) (Li et al. 2010). We induce a distribution over possible *methods* by using a sum of 3 different heuristics, post an n-step look-ahead in the decomposition tree.

1. D_{m_i} - Distance to goal from the current state, post look-ahead
2. C_{m_i} - Cost of the partial plan after look-ahead.
3. $A_{m_i}^{(pref)}$ - Adherence to the current set of preferences.

Assume $M_\tau = \{m_i\}_{i=1}^k$ be the set of k legal methods that can decompose a given task τ. For every method m_i, roll-out of decompositions is perfomed till a depth cutoff n (tunable parameter). Note that the state and partial plan may also have changed after rollout if primitive tasks were encountered. Post roll-out, we calculate the quantities D_{m_i}, C_{m_i} and $A_{m_i}^{(pref)}$. However, D_{m_i} and C_{m_i} are quantities we wish to minimize and $A_{m_i}^{(pref)}$ is somthing we wish to maximize. Hence, to establish uniformity one of them needs to be inverted. In our case, we use the inverse of D_{m_i} and C_{m_i}. So the final score of m_i becomes,

$$S_{m_i} = inv\,(D_{m_i} + C_{m_i}) + A_{m_i}^{(pref)} \qquad (1)$$

Thus for a given task we get k-length vector of scores $[S_{m_1}, \dots, S_{m_k}]$. We convert this set of scores into a probability distribution $\mathcal{P}_{M_\tau} = [p_{m_i}, \dots, p_{m_k}]$. To get the uncertainty we measure entropy of the distribution,

$$U(M_\tau) = \sum_{m \in M_\tau} p_m . \log(1/p_m) \qquad (2)$$

The heuristic has a few interesting properties - (1) Plan Cost C_{m_i} is equal to plan length when each action is of unit cost. Over estimation is impossible since partial plans are stored at each step. (2) Distance to goal D_{m_i} is the only domain-dependent component of the entire framework. Domain independence is achievable just by counting the number of goal state literals that are unsatisfied by the current state, but that is an extremely rough estimate. This is an admissible heuristic since we always take the least number of steps a particular object might take to be in its final state. (3) $A_{m_i}^{(pref)}$ is just an adherence metric to current preferences. Unlike, Sohrabi's work (2009), there is no weight on the preferences. Hence the final score is an admissible heuristic owing to its components being admissible.

Uncertainty based active preference elicitation

Given that we can measure uncertainty over the *methods* that decompose a particular task, our active preference elicitation framework *UAct* will query a human for suggestions/preferences when the uncertainty is above a certain threshold ϵ. The architecture is presented in figure 3. In brief, the HTN planner picks a task τ to process. Then, $\forall m \in M_\tau$ where M_τ is the set of possible legal methods that can decompose τ it rolls-out the decomposition to a depth of n. The roll-out policy is uniform random. Then it gets the score for every m, calculates the uncertainty and decides whether to ask for preference. Then it poses the following query:

```
<Planner>:Trying to solve Task T (Ground task atom)
    Decomposition options:
    x : [(subtask1),(subtask2),...,(subtaskj)]
    y : [(subtask1),(subtask2),...,(subtaskj)]
    z : [(subtask1),(subtask2),...,(subtaskj)]
    Choose or provide any other suggestion.
```

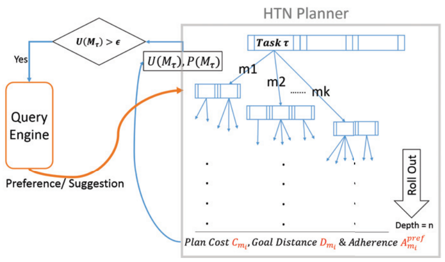


Figure 3: Architecture of *UAct* Framework

x, y, z are option numbers and j is the number of subtasks that might vary across options. Preference can be a particular method choice or text (with limited vocabulary), which is parsed into a horn clause, since they closely model IF-THEN rules, and the method satisfying the clause is executed. If that does not work, method with highest score is used. The system is able to solicit suggestions and preferences in different forms from the humans,

- Low level preferences - that asks the planner to perform a particular action. *(Example: "Move the seven of hearts to the top of the three of clubs")*. This move is executed if the planner finds it legal.
- High level preferences - that suggests the planner on what the priorities are and how the game should be played. *(Example: "Try getting rid of Aces first")*. These kinds of preferences end up pruning the task decomposition tree.
- Decomposition choices - where the human chooses among potential decomposition alternatives. This results in hard pruning of the decomposition tree.

The formal algorithm for *UAct* is present in Algorithm 1

Implementation & Experiments

We have used the Java based implementation of the SHOP2 (Nau et al. 2003) architecture (JSHOP) as the base HTN planner. We have extended it to - (1) expose the *task* being processed at the current step to the human, (2) find and evaluate all viable alternatives/methods for decomposing the same and (3) elicit human feedback/preference based on uncertainty over the alternatives. All the extensions are implemented in Java. In our experiments we aim to answer the following questions, **(Q1):** How does actively seeking preference affect solvability, against baselines? **(Q2):** Does our approach affect performance in terms of optimality of solutions? and **(Q3):** Ease of use for a human giving suggestion/preference as opposed to other techniques?

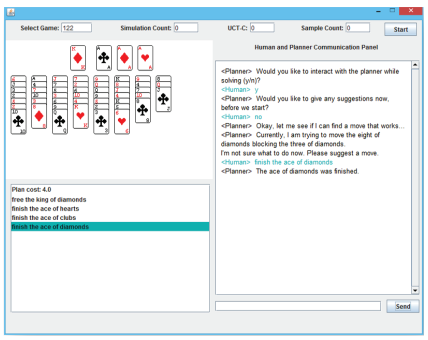
We compared *UAct* against several alternate approaches for preference elicitation such as (1) Upfront preference encoded as knowledge (2) Random querying with probability $\mathcal{P}$, which is tunable (3) Planning with no preferences. The last one, though seems to be trivial as a baseline, yet is necessary since we do not assume the expertise of the human. Consequently, human preferences can potentially push the search process away from optimality. Hence, in some problems planning with preferences performs worse than without any. While our longer-term plan is to evaluate our strategy in several well known planning domains, we now present results for one of them, namely Freecell.

Algorithm 1 Uncertainty based active pref (*UAct*)

```
1:  procedure UACT(Task $\tau$, Current State $\mathfrak{S}$)
2:      Input: Task $\tau$
3:      Output: Query $Q$
4:      Initialization: $M_\tau \leftarrow methods(\tau, \mathfrak{S})$
5:      $U(M_\tau) \leftarrow EntropyCalc(M_\tau, \mathfrak{S}, n)$
6:      if $U(M_\tau) > \epsilon$ then
7:          $Q \leftarrow \tau + M_\tau$(options) + "Any suggestion?"
8:      else
9:          $Q \leftarrow null$
10:     end if
11: return $Q$
12: end procedure
13: ————————————————————————
14: procedure ENTROPYCALC(Methods $M_\tau$, State $\mathfrak{S}$, RollOut Depth $n$)
15:     Initialize: $S(M_\tau)$ a list for scores of methods
16:     for each m in $M_\tau$ do
17:         while $depth \leq n$ do
18:             Decompose [update $CurrPlan$ and $\mathfrak{S}$]
19:             $depth \leftarrow depth + 1$
20:         end while
21:         $C \leftarrow cost(CurrPlan)$ ; $D \leftarrow \delta(\mathfrak{S}, goal)$
22:         $A \leftarrow n_t - n_F$
23:         $put\,(S(M_\tau), m : inv(D + C) + A)$
24:     end for
25:     Compute $\mathcal{P}(M_\tau)$
26:     Compute $U(M_\tau) \leftarrow \sum_{i=1}^{k} p_{m_i} . \log(1/p_{m_i})$
27: return $U(M_\tau)$
28: end procedure
```

Results with Freecell We chose Freecell owing to its popularity in several International Planning Competitions and its interesting state and action dynamics. Also, being a well known game, any arbitrary human can solicit potentially useful suggestion/preference without requiring to be a domain expert. Briefly, a complete freecell game has a *tableau* with 8 columns which can hold at most 7 cards (face up), 4 *home cells* one for each suit, where the cards are supposed to go in order of their ranks (Ace to King) and 4 *free cells*, which act as the overflow spaces that are used for game playing. A card can be moved between columns, from columns to free cell or vise versa. A card can only be moved to the home cell (finished) if the immediately lower card of the same suit has already been finished. Several other constraints/rules exist that determine legal actions and are encoded in the domain definition. At the initial state, a *deal* is chosen and cards are laid out in the columns of the tableau. The goal, always, is to finish the highest ranked card in every suit (the kings in a complete domain) Games can be relaxed either by increasing the number of free cells or by taking less that 13 cards per suit. Note that the planner and the preference framework are independent of such customizations, since the game itself is encoded as a set of ground literals.

To evaluate the effectiveness of our approach against all baselines, we experimented with 20 different problems

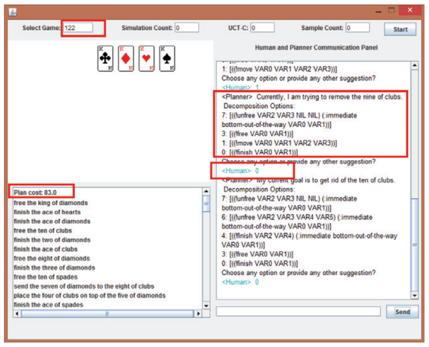


```
Select Game: 122   Simulation Count: 0   UCT-C: 0   Sample Count: 0   Start

Human and Planner Communication Panel
<Planner>  Would you like to interact with the planner while solving (y/n)?
<Human>  y
<Planner>  Would you like to give any suggestions now, before we start?
<Human>  no
<Planner>  Okay, let me see if I can find a move that works...
<Planner>  Currently, I am trying to move the eight of diamonds blocking the three of diamonds.
I'm not sure what to do now. Please suggest a move.
<Human>  finish the ace of diamonds
<Planner>  The ace of diamonds was finished.

Plan cost: 4.0
free the king of diamonds
finish the ace of hearts
finish the ace of clubs
finish the ace of diamonds
```

(a) With active preference

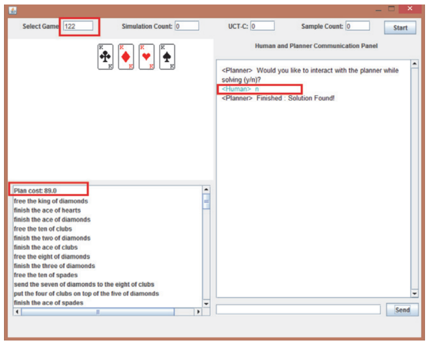


```
Select Game: 122   Simulation Count: 0   UCT-C: 0   Sample Count: 0   Start

Human and Planner Communication Panel
1: [((!move VAR0 VAR1 VAR2 VAR3))]
Choose any option or provide any other suggestion?
<Human>  1
<Planner>  Currently, I am trying to remove the nine of clubs.
 Decomposition Options:
7: [((!unfree VAR2 VAR3 NIL NIL) (:immediate bottom-out-of-the-way VAR0 VAR1))]
3: [((!free VAR0 VAR1))]
1: [((!move VAR0 VAR1 VAR2 VAR3))]
0: [((!finish VAR0 VAR1))]
Choose any option or provide any other suggestion?
<Human>  0
<Planner>  My current goal is to get rid of the ten of clubs.
 Decomposition Options:
7: [((!unfree VAR2 VAR3 NIL NIL) (:immediate bottom-out-of-the-way VAR0 VAR1))]
6: [((!unfree VAR2 VAR3 VAR4 VAR5) (:immediate bottom-out-of-the-way VAR0 VAR1))]
4: [((!finish VAR2 VAR4) (:immediate bottom-out-of-the-way VAR0 VAR1))]
3: [((!free VAR0 VAR1))]
0: [((!finish VAR0 VAR1))]
Choose any option or provide any other suggestion?
<Human>  0

Plan cost: 83.0
free the king of diamonds
finish the ace of hearts
finish the ace of diamonds
free the ten of clubs
finish the two of diamonds
finish the ace of clubs
free the eight of diamonds
finish the three of diamonds
free the ten of spades
send the seven of diamonds to the eight of clubs
place the four of clubs on top of the five of diamonds
finish the ace of spades
```

(b) With active preference (After completion)



```
Select Game 122   Simulation Count: 0   UCT-C: 0   Sample Count: 0   Start

Human and Planner Communication Panel
<Planner>  Would you like to interact with the planner while solving (y/n)?
<Human>  n
<Planner>  Finished : Solution Found!

Plan cost: 89.0
free the king of diamonds
finish the ace of hearts
finish the ace of diamonds
free the ten of clubs
finish the two of diamonds
finish the ace of clubs
free the eight of diamonds
finish the three of diamonds
free the ten of spades
send the seven of diamonds to the eight of clubs
put the four of clubs on top of the five of diamonds
finish the ace of spades
```

(c) No preference (After completion)

Figure 4: Planning on freecell (with and without preference)

(freecell games), 10 complete and 10 relaxed (5 with 8 cards per suit and 5 free cells and the other 5, 10 cards per suit with 4 free cells). Figure 5 shows a comparison of different approaches for the percentage of problems solved (plan is generated), in a given time constraint. We chose the time bound to be 10 minutes for every problem. The green bar in the plot proves that our method allows for fast plan generation in almost 100% of the problems in a given time constraint.

The time constraint is reasonable since, (1) planning problems are motivated by the need for *fast* generation of optimal or near-optimal solutions, and (2) relaxing the time constraint may increase performance of the baseline approaches, but will similarly push ours to 100%. Our approach never falls short in terms of efficiency since active preferences always prunes the search space effectively, thus successfully answering **Q1**.

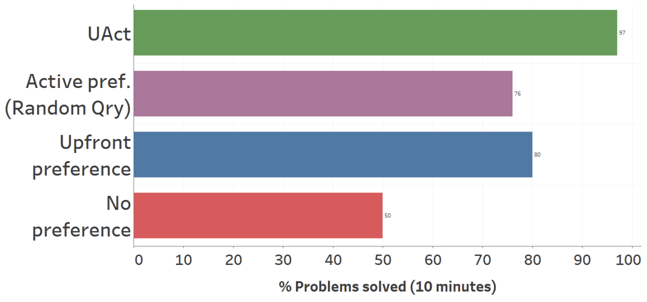


Figure 5: % Problems solved in 10 Minutes

We further compared the performance of the several approaches in terms of the optimality of the generated plans, using only 50% of the problems - the ones that are solvable without any preference. Table 1 compares the our approach, *(UAct)* against all other baselines. The values indicate the percentage of problems in which our proposed approach produced better (or at least equivalent) plans than the approach compared against. Here better implies lower plan length/cost. Clearly our approach outperforms baselines in terms of optimality of solutions, hence successfully answering **Q2**. As an example, figures 4(b) & 4(c) demonstrate that, for a certain problem (deal 122), planning with no preference performs worse ($plancost = 89$) than our framework ($plancost = 83$)

	% ≥
UAct vs No Pref	80
UAct vs Active Pref (Random)	50
UAct vs Upfront Preference	60

Table 1: Percentage of problems, in freecell, in which one performed better than the other (Lower plan length = Better)

Answering **Q3** is trickier. Evaluating and quantifying ease of preference giving needs an extensive survey. We may attempt that at a later stage, but presently, we try to argue our case with the interface that we built. Figures 4(a) & 4(b) demonstrate how a human is able to provide different types of suggestions as discussed in earlier sections. More importantly the human is not burdened with the preference giving process. The planner decides where it needs one and queries the human. Unlike, Sohrabi et al.'s (2009) work where all the preferences are encoded upfront into the domain and the problem descriptions, our framework may start with no preference at all and query for it as the planning proceeds.

Observe that upfront preference approach is the closest in performance and may prove to be as good as ours with careful encoding. However, exhaustively encoding knowledge without observing planner's current state is difficult even for an expert, since (s)he is unaware of the parts of the state space that require additional knowledge (Odom and Natarajan 2016). Consequently, active elicitation, where human can decide on what to suggest based on the query and state, is provably more effective.

Conclusion

We presented our work on active preference elicitation in Hierarchical Task Network planning. The initial results demonstrate that our approach performs better compared to standard baselines. Evaluation on several other domains and a robust collaborative planning framework are our future research directions.

Acknowledgement

We gratefully acknowledge the support of DEFT program Contract FA8750-13-2-0039 and CwC Program Contract W911NF-15-1-0461 with the US Defense Advanced Research Projects Agency (DARPA) and the Army Research Office (ARO). Any opinions, findings and conclusion or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the DARPA, ARO or the US government.

References

Allen, J., and Ferguson, G. 2002. Human-machine collaborative planning. In *Proceedings of the Third International NASA Workshop on Planning and Scheduling for Space*.

Boutilier, C.; Brafman, R.; Geib, C.; and Poole, D. 1997. A constraint-based approach to preference elicitation and decision making. In *AAAI spring symposium on qualitative decision theory*.

Boutilier, C.; Brafman, R. I.; Domshlak, C.; Hoos, H. H.; and Poole, D. 2004. Preference-based constrained optimization with cp-nets. *Computational Intelligence* 20(2):137–157.

Boutilier, C.; Dean, T.; and Hanks, S. 1995. Planning under uncertainty: Structural assumptions and computational leverage. In *Proceedings of the Second European Workshop on Planning*.

Brafman, R. I., and Chernyavsky, Y. 2005. Planning with goal preferences and constraints. In *ICAPS*, 182–191.

Dean, T.; Kaelbling, L. P.; Kirman, J.; and Nicholson, A. 1995. Planning under time constraints in stochastic domains. *Artificial Intelligence* 76(1):35–74.

Ghallab, M.; Nau, D.; and Traverso, P. 2004. *Automated planning: theory & practice*. Elsevier.

Huang, Y.-C.; Selman, B.; Kautz, H.; et al. 1999. Control knowledge in planning: benefits and tradeoffs. In *AAAI/IAAI*, 511–517.

Judah, K.; Roy, S.; Fern, A.; and Dietterich, T. G. 2010. Reinforcement learning via practice and critique advice. In *AAAI*.

Kunapuli, G.; Odom, P.; Shavlik, J.; and Natarajan, S. 2013. Guiding autonomous agents to better behaviors through human advice. In *ICDM*.

Li, N.; Cushing, W.; Kambhampati, S.; and Yoon, S. 2010. Learning probabilistic hierarchical task networks to capture user preferences. *arXiv preprint arXiv:1006.0274*.

Maclin, R., and Shavlik, J. W. 1994. Incorporating advice into agents that learn from reinforcements. In *AAAI*.

Maclin, R., and Shavlik, J. W. 1996. Creating advice-taking reinforcement learners. *Machine Learning* 22(1):251–281.

Myers, K. L. 1996. Advisable planning systems. *Advanced Planning Technology* 206–209.

Natarajan, S., and Tadepalli, P. 2005. Dynamic Preferences in Multi-Criteria Reinforcement Learning. In *ICML*.

Nau, D. S.; Au, T.-C.; Ilghami, O.; Kuter, U.; Murdock, J. W.; Wu, D.; and Yaman, F. 2003. Shop2: An htn planning system. *J. Artif. Intell. Res. (JAIR)* 20:379–404.

Odom, P., and Natarajan, S. 2016. Active advice seeking for inverse reinforcement learning. In *AAMAS*.

Sohrabi, S., and McIlraith, S. A. 2008. On planning with preferences in htn. In *(NMR)*.

Sohrabi, S.; Baier, J. A.; and McIlraith, S. A. 2009. Htn planning with preferences. In *IJCAI*.

Tan, S.-W., and Pearl, J. 1994. Specification and evaluation of preferences for planning under uncertainty. In *KR*.

Torrey, L.; Walker, T.; Shavlik, J.; and Maclin, R. 2005. Using advice to transfer knowledge acquired in one reinforcement learning task to another. In *ECML*, 412–424.