`AI-MIX`: Using Automated Planners to Steer Human Workers Towards Better Crowdsourced Plans

Lydia Manikonda[*]
Advisor: Prof. Subbarao Kambhampati
Department of Computer Science
Arizona State University
Tempe, Arizona 85281
Email: *lmanikon@asu.edu*

Abstract

One subclass of human computation applications are those directed at tasks that involve planning (e.g. tour planning) and scheduling (e.g. conference scheduling). Existing literature on these systems show that even primitive automated oversight on human contributors help in significantly improving the effectiveness of humans/crowd. In this paper, we explain how the automated oversight used in such systems can be viewed as a primitive automated planner, and outline several opportunities for more sophisticated automated planning in effectively steering crowdsourced planning. Adapting the current planning technology requires overcoming the mismatch between the capabilities of human workers and automated planners. To overcome this mismatch, two important challenges are to be addressed. They are (i) *interpreting* the inputs of human workers and (ii) *steering* or critiquing the plans produced by the human workers provided the *incomplete* domain and preference models. We discuss these challenges and describe our initial attempt at tackling them. Specifically, we describe our ongoing work **`AI-MIX`**, a tour-plan generation system that uses automated checks and alerts to help improve the quality of plans created by human workers.

Keywords. Crowdsourcing, Automated Planning, Artificial Intelligence

1 Motivation

In recent times, there has been a significant interest in crowdsourcing and human computation. In solving computationally hard problems – especially those that require input from humans, or for which the complete model is not known – human computation has emerged as a powerful and inexpensive approach. One such core class of problems is *planning* (e.g. tour planning) (Manikonda et al. 2014; Talamadupula et al. 2013). Most of this work appears outside the traditional automated planning forums, and it is not clear if automated planning plays any role in these "human computation" systems.

Interestingly, literature on these systems shows that even primitive forms of automated instructions to steer the human contributors have helped the effectiveness of humans/crowd significantly. Several recent efforts have started looking at crowd-sourced planning tasks (Law and Zhang 2011; Zhang et al. 2012; 2013). We observe that in most of these existing systems, the workers are steered by primitive automated components that merely enforce checks and ensure satisfaction of simple constraints. Encouragingly, experiments show that this planning approach where humans and the system work together in a tight coupled way can improve plan quality, for little to no investment in terms of cost and time.

2 Background & Related Work

The key question is: *is it possible to improve the effectiveness of crowdsourced planning even further by using more sophisticated automated planning technologies?* It is reasonable to expect that a more sophisticated automated planner can do a much better job of steering the crowd. Indeed, work such as (Law and Zhang 2011) and (Zhang et al. 2012) is replete with hopeful references to the automated planning literature. There exists a vibrant body of literature on automated plan generation, and automated planners have long tolerated humans in their decision cycle – be it mixed initiative planning (Ferguson, Allen, and Miller 1996) or planning for teaming (Talamadupula et al. 2010). The context of crowdsourced planning scenarios, however, introduces a *reversed mixed initiative planning* problem – the planner must act as a guide to the humans, who are doing the actual planning. The humans in question can be either experts who have a stake in the plan that is eventually created, or crowd workers demonstrating collective intelligence.

Here, we present our ongoing work on **`AI-MIX`** (Automated Improvement of Mixed Initiative eXperiences), a new system (Manikonda et al. 2014) which implements a general architecture for human computation systems aimed at planning and scheduling tasks. **`AI-MIX`** foregrounds the types of roles an automated planner can play in such systems, and the challenges involved in facilitating those roles. The most critical challenges include:

Interpretation: Understanding the requester's goals as well as the crowd's plans from semi-structured or unstructured natural language input.

Steering with Incompleteness: Guiding the collaborative plan generation process with the use of incomplete models of the scenario dynamics and preferences.

[*]Part of this work was done in collaboration with Tathagata Chakraborti, Sushovan De and Kartik Talamadupula and got published at IAAI-2014 (Manikonda et al. 2014)

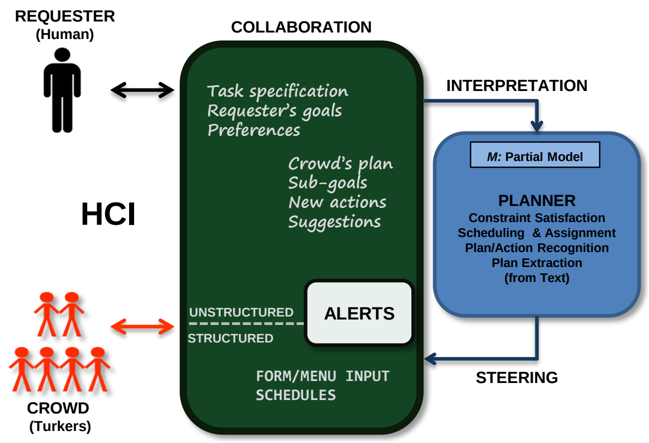


Figure 1: A generalized architecture for crowdsourced planning systems.

The *interpretation* challenge arises because human workers find it most convenient to exchange / refine plans expressed in a representation as close to natural language as possible, while automated planners typically operate on more structured plans and actions. The challenges in *steering* are motivated by the fact that an automated planner operating in a crowdsourced planning scenario cannot possibly be expected to have a complete model of the domain and the preferences; if it does, then there is little need or justification for using human workers! Both these challenges are further complicated by the fact that the (implicit) models used by the human workers and the automated planner are very likely to differ in many ways, making it challenging for the planner to critique the plans being developed by the human workers.

3 Proposed Research

3.1 Planning for Crowdsourced Planning

The crowdsourced planning problem involves returning a plan as a solution to a task, usually specified by a user called the *requester*. The requester provides a high-level description of the task – most often in natural language – which is then forwarded to the *turkers*. The turkers can perform various roles, including breaking down the high-level task description into more formal and achievable sub-goals (Law and Zhang 2011), or adding actions into the plan that support those sub-goals (Zhang et al. 2012). The *planner* is the automated component of the system, and it performs various tasks ranging from constraint checking, to optimization and scheduling, and plan recognition. The entire planning process must itself be iterative, proceeding in several rounds which serve to refine the goals, preferences and constraints further until a satisfactory plan is found. A general architecture for solving this crowdsourced planning problem is depicted in Figure 1.

Roles of the Planner The planning module, or the automated component of the system, can provide varying levels of support. It accepts both the sub-goals S_G, and crowd's plan P_C, as input from the turkers. This module analyzes the current plan generated by the crowd, as well as the sub-goals, and determines constraint and precondition violations according to the model M of the task that it has. The planner's job is to steer the crowd towards more effective plan generation.

However, the three main actors – turkers, requester, and planner – need a common space in which to interact and exchange information. This is achieved through a common interactive space – the *Collaborative Blackboard* (DBb) – as shown in Figure 1. The DBb acts as a collaborative space where information related to the task as well as the plan that is currently being generated is stored, and exchanged between the various system components. In contrast to the turkers, the planner cannot hope for very complex, task-specific models, mostly due to the difficulty of creating such models. Instead, a planner's strong-suit is to automate and speed-up the checking of plans against whatever knowledge it *does* have.

3.2 Challenges

From the architecture described in Figure 1, it is fairly obvious that a planner (automated system) would interact with the rest of the system to perform one of two tasks: (1) **interpretation** and (2) **steering**. Interpretation is required for the planner to inform itself about what the crowd *is* doing; steering is required for the planner to tell the crowd what they *should* be doing.

Interpretation of the Crowd's Evolving Plan The planner must interpret the information that comes from the requester, and from the crowd, in order to act on that information. There are two ways in which the planner can ensure that it is able to understand that information:

I. Force Structure The system can enforce a predetermined structure on the input from both the requester, and the crowd. This can by itself be seen as part of the model M_p, since the planner has a clear idea about what kind of information can be expected through what channels. The obvious disadvantage is that it reduces flexibility for the turkers. In the tour planning scenario, for example, we might force the requester to number his/her goals, and force the turkers to explicitly state which goals their proposed plan aims to handle (c.f. (Zhang et al. 2012)). The turkers could also be required to add other structured attributes to their plans such as the duration and cost of various activities (actions) that are part of the plan.

II. Extract Structure The planner can also *extract* structure from the turker inputs to look for specific action descriptions that are part of the planner's model M_P, in order to understand what a specific plan is looking to achieve. Although this problem has connections to plan recognition (Kautz and Allen 1986; Ramírez and Geffner 2010), it is significantly harder as it needs to recognize plans not from actions, but rather textual descriptions. Thus it can involve first recognizing actions and their ordering from text, and then recognizing plans in terms of those actions. Unlike traditional plan recognition that starts from observed plan traces in terms of actions or actions and states, the interpretation involves first extracting the plan traces. Such recognition is further complicated by the impedance mismatch between the (implicit) planning models used by the human workers, and the model available to the planner.

Our current system uses both the techniques described above to gather relevant information from the requester and the turkers. The requester provides structured input that lists their constraints as well as goals (and optionally cost and duration constraints), and can also provide a free unstructured text description for the task. The turkers in turn also provide semi-structured data - they are given fields for activity title, description, cost and duration. The turkers can also enter free text descriptions of their suggestions; the system can then automatically extract relevant actions by using natural language processing techniques to match the input against the planner's model M_P.

Steering the Crowd's Plan The planner can steer the turkers by offering helpful suggestions, alerts, and perhaps even its own plan. There are two main kinds of feedback an automated planner can provide to the human workers:

I. Constraint Checking One of the simplest ways of generating helpful suggestions for the crowd is to check for quantitative constraints imposed by the requester that are violated in the suggested activities. In terms of the tour planning scenario, this includes: (i) cost of a particular activity; and (ii) the approximate duration of an activity. If the requester provides any such preferences, our system is able to check if they are satisfied by the crowd's inputs.

II. Constructive Critiques Once the planner has some knowledge about the plan that the turkers are trying to propose (using the extraction and recognition methods described above), it can also try to actively help the creation and refinement of that plan by offering suggestions as part of the alerts. These suggestions can vary depending on the depth of the planner's model. Some examples include: (i) simple notifications of constraint violations, as outlined previously; (ii) plan critiques (such as suggestions on the order of actions in the plan and even what actions must be present); (iii) new plans or plan fragments because they satisfy the requester's stated preferences or constraints better; (iv) new ways of decomposing the current plan (Nau et al. 2003); and (v) new ways of decomposing the set of goals S_G.

3.3 Current System Description

The following section describes in detail the **AI-MIX** system that was deployed on Amazon's Mechanical Turk platform to engage the turkers in the tour planning task. The system is similar to Mobi (Zhang et al. 2012) in terms of the types of inputs it can handle and the constraint and quantity checks that it can provide. However, instead of using structured input, which severely restricts crowd turkers and limits the scope of their contributions, our system is able to parse natural language from user inputs and reference it against relevant actions in a domain model. This enables more meaningful feedback and helps provide a more comprehensive tour description.

Interface for Turkers The main **AI-MIX** interface, shown in Figure 2, contains the task description (as provided by the requester) and a section that lists instructions for successfully submitting a HIT on Amazon MTurk. The remaining components, arranged by their labels in the figure, are:

1. **Requester Specification**: This is a brief description of the preferences, followed by a list of activities they want to do as part of the tour, each accompanied by a suitable hashtag. For example, the requester might include one dinner activity and associate it with the tag #dinner. These tags are used internally by the system to map turker suggestions to specific tasks.
2. **Turker Inputs**: Turkers can choose to input one of two kinds of suggestions: (i) a new action to satisfy an existing to-do item; or (ii) a critique of an existing plan activity (action).
3. **Turker Responses**: The "Existing Activities" box displays a full list of the current activities that are part of the plan. New turkers may look at the contents of this box in order to establish the current state of the plan. This component corresponds to the *Distributed Blackboard* mentioned in Section 3.1.

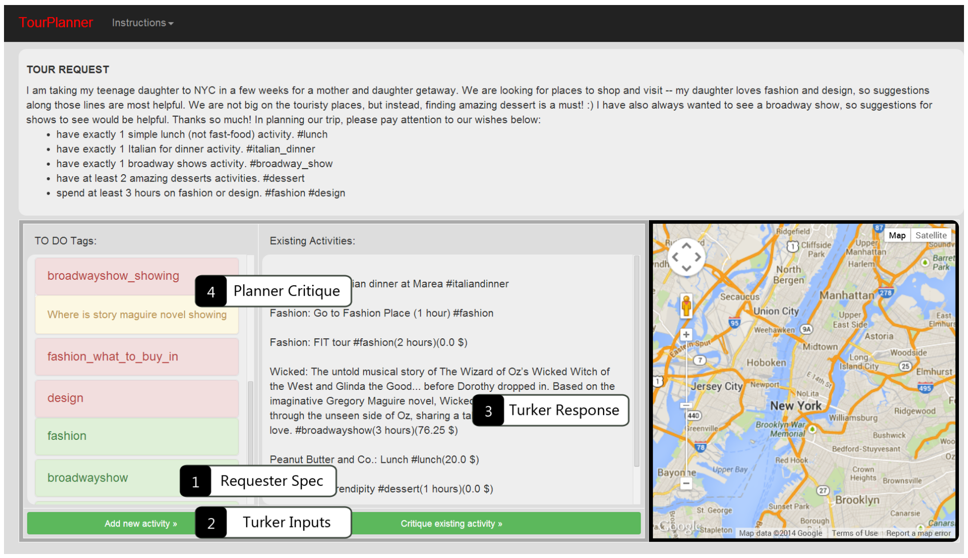
Finally, the right hand portion of the interface consists of a map, which can be used by turkers to find nearby points of interest, infer routes of travel or the feasibility of existing suggestions, or even discover new activities that may satisfy some outstanding tags.

Activity Addition Turkers may choose to add as many new activities as they like. Each new activity is associated with one of the to-do tags. After each activity is submitted, a quantitative analysis is performed where the activity is (i) checked for possible constraint (duration or cost) violations; or (ii) critiqued the planner.

Action Extraction To facilitate the extraction of meaning from the turker generated activities, the system performs parts of speech (PoS) tagging on the activities to identify the name of the activity as well as the places that turkers are referring to; currently, we assign the *verb* and *noun* parts of the tagger's output to these respectively. We used the *Stanford Log-Linear Part-of-Speech* tagger (Toutanova et al. 2003).

Sub-Goal Generation **AI-MIX** uses the same tags used by turkers while inputting activities in order to determine whether the planner has additional subgoal annotations on that activity. To facilitate this, the planner uses a primitive PDDL (McDermott et al. 1998) domain description of general activities that may be used in a tour-planning applications – this description corresponds to the *planner model* M_P introduced previously. Examples of actions in M_P include activities such as `visit`, `lunch`, `shop` etc. Each action comes with a list of synonyms, which help the planner in identifying similar activities. Each action also comes with some generic preconditions. When the planner determines that a turker generated activity matches one of the actions from its model, it generates sub-goals to be added as to-do items back in the interface based on the preconditions of that action.

Constraint Checking In addition to generating sub-goals for existing activities, our system also automatically checks if constraints on duration and cost that are given by the requester are being met by the crowd's plan. If these constraints are violated, then the violation is automatically added to the to-do stream of the interface, along with a description of the constraint that was violated. Turkers can then choose to add an action that resolves this to-do item using the normal procedure.



TOUR REQUEST

I am taking my teenage daughter to NYC in a few weeks for a mother and daughter getaway. We are looking for places to shop and visit -- my daughter loves fashion and design, so suggestions along those lines are most helpful. We are not big on the touristy places, but instead, finding amazing dessert is a must! :) I have also always wanted to see a broadway show, so suggestions for shows to see would be helpful. Thanks so much! In planning our trip, please pay attention to our wishes below:

- have exactly 1 simple lunch (not fast-food) activity. #lunch
- have exactly 1 Italian for dinner activity. #italian_dinner
- have exactly 1 broadway shows activity. #broadway_show
- have at least 2 amazing desserts activities. #dessert
- spend at least 3 hours on fashion or design. #fashion #design

TO DO Tags:

broadwayshow_showing

Where is story maguire novel showing

fashion_what_to_buy_in

design

fashion

broadwayshow

Existing Activities:

ian dinner at Marea #italiandinner

Fashion: Go to Fashion Place (1 hour) #fashion

Fashion: FIT tour #fashion(2 hours)(0.0 $)

Wicked: The untold musical story of The Wizard of Oz's Wicked Witch of the West and Glinda the Good... before Dorothy dropped in. Based on the imaginative Gregory Maguire novel, Wicke through the unseen side of Oz, sharing a ta love. #broadwayshow(3 hours)(76.25 $)

Peanut Butter and Co.: Lunch #lunch(20.0 $)

rendipity #dessert(1 hours)(0.0 $)

Figure 2: The tour planner interface of the **AI−MIX** system on MTurk.

Adding Turker Critiques The turkers can also choose to add *critiques* of actions in the existing plan, instead of adding actions that satisfy existing to-do items. The turkers click on an existing activity, and enter the note or critique in a text box provided. Turkers are free to add as many critiques as they want.

A video run-through of our system can be found at the following URL: `http://youtu.be/73g3yHClx90`. Also, this system is available on the Amazon EC2 server at `http://bit.ly/1qD539I`.

3.4 Experiments

Experimental Setup For our study, HITs were made available to all US residents with a HIT approval rate greater than 50%. Turkers were paid 20 cents for each HIT, and each turker could submit 10 HITs per task. We use tour planning scenarios for six major US cities, reused from the Mobi system's evaluation (Zhang et al. 2012). To measure the impact of automated critiquing on the generated plans, we compare the results from two experimental conditions:

C1: Turkers quantify their suggestions in terms of cost and duration, and the system checks these constraints for violations with respect to the requester demands.

C2: In addition to C1, we process free-form text from turker input, and extract actions to match with our planning model in order to generate alerts for sub-goals and missing preconditions.

C1 was compared to the proposed approach, C2, separately. Each set was uploaded at the same time, with the same task description and HIT parameters. In the first run, C1 and C2 were compared on 6 scenarios and were given 2 days before the HITs were expired. The interface for both C1 and C2 is identical to eliminate any bias. In sum, we had more than 150 turkers who responded to our HITs.

Generated Tour Plan Quality We see that the quality of the plans, in terms of detail and description, seems to increase in C2, since we now have users responding to planner critiques to further qualify suggested activities. For example, a turker suggested "not really fun, long lines and can not even go in and browse around" in response to a planner generated tag (related to a "fun club" activity suggested previously), while another suggested a "steamer" in response to a planner alert about "what to eat for lunch". A comparison between the plans generated by C1 and C2 (for New York City) is given in Table 1. This seems to indicate that including a domain description in addition to the simplistic constraint checks increases the plan quality.

Role Played by the Planner Module We now look at some statistics that capture the role played by the planning module in the tasks. We received a total of 31 new activity suggestions from turkers, of which 5 violated quantity constraints. The C2 interface attracted 39 responses, compared to 28 for C1, which may indicate that the planner tags encouraged turker participation.

Note that in the **AI−MIX** interface, there is no perceptual difference between the critiques generated by the planner and the critiques suggested by humans. With this in mind, there were 8 flaws pointed out by humans, but none were acted upon by other turkers; the planner on the other hand generated 45 critiques, and 7 were acted upon and fixed by turkers. This seems to indicate that turkers consider the planner's critiques more instrumental to the generation of a high quality plan than those suggested by other turkers. Though these results are not entirely conclusive, there is enough evidence to suggest that the presence of an automated critiquing system does help to engage and guide the focus of the crowd.

4 Research Directions

While the performance of the current implementation of **AI−MIX** is promising, there do exist several avenues for further research. Two directions that we are currently pursuing include – plan extraction and model improvement. We briefly discuss these directions below.

Show: Go to TKTS half ticket discount booth. You have to stand in line early but it's an authentic nyc experience #show(3 hours)(200.0 $)
Show: Go to show #show(3 hours)(200.0 $)
Show: ABSOLUTELY CANNOT go wrong with Phantom of the Opera #show(3 hours)(200.0 $)
Lunch: Alice's Tea Cup #lunch(20.0 $)
Design: Walk around the Garment District (go into shops) just south of Times Square. They often print their own fabrics. #design(2 hours)(0.0 $)
Dessert: Serendipity #dessert(1 hours)(10.0 $)

piccolo angolo: Italian in the Village - real deal #italiandinner(2 hours)(60.0 $)
Lombardi's Pizza: #italian_dinner #italiandinner_todo1
Ice Cream: http://www.chinatownicecreamfactory.com/ #italiandinner_todo0
#lunch: Mangia Organics #lunch_todo0
watch Wicked (musical): Do watch Wicked the musical. It's a fantastic show and one of the most popular on Broadway right now! #broadwayshow(3 hours)(150.0 $)
watch How to Succeed in Business: Also a great show, a little less grand than Wicked. #broadwayshow(3 hours)(150.0 $)
Activity Steamer: #lunch #lunch_todo1
Paradis To-Go: Turkey & Gruyere is pretty delicious. The menu is simple, affordable, but certainly worth the time #lunch(1 hours)(10.0 $)
cupcakes!: Magnolia Bakery on Bleecker in the Village #dessert(1 hours)(10.0 $)

Table 1: Sample activity suggestions from turkers for the two conditions: C1 (top) and C2 (bottom).

4.1 Plan Extraction

The current method for action extraction is a naive approach that uses only the first verb to extract the candidate action from the turkers' input. As we realized from the initial set of experiments that this basic framework is able to extract the actions well, but if the input given by the user consists of more than one action, the planner won't be able to extract all the actions of this plan. This means subgoals may not be satisfied even if they should have been if the full set of actions were extracted. In order to address this, we are currently focusing on adapting specific approaches proposed by (Zhang et al. 2013; Kim, Chacha, and Shah 2013; Addis and Borrajo 2011). These methods extract structured plans by using crowdsourcing, temporal information or building probabilistic generative models with logical plan validation.

4.2 Model Improvement

The current plan steering differs in significant ways from the traditional plan synthesis and plan critiquing. This is due to the incompleteness of the domain model and the requester's preferences available to the planner. Traditional techniques view planning as producing a provably correct course of action. But what may be seen as a correct or optimal plan because of the planner's incomplete domain model can be a undesirable one from the requester's point of view. Interestingly, it is possible to improve the completeness of the domain model of a planner over time. Considering this, we are interested to see if the existing work on learning domain models (Yang, Wu, and Jiang 2007; Blythe 2005) can be adapted to allow learning from observing the plans suggested by the crowd.

References

Addis, A., and Borrajo, D. 2011. From unstructured web knowledge to plan descriptions. In *Information Retrieval and Mining in Distributed Environments*, volume 324. 41–59.

Blythe, J. 2005. Task learning by instruction in tailor. In *IUI*, 191–198.

Ferguson, G.; Allen, J.; and Miller, B. 1996. Trains-95: Towards a mixed-initiative planning assistant. In *Proceedings of Third Conference on Artificial Intelligence Planning Systems*, 70–77.

Kautz, H., and Allen, J. F. 1986. Generalized plan recognition. In *Proc. of 5th National Conference on AI*, volume 1, 32–37.

Kim, B.; Chacha, C. M.; and Shah, J. A. 2013. Inferring robot task plans from human team meetings: A generative modeling approach with logic-based prior. In *AAAI*.

Law, E., and Zhang, H. 2011. Towards large-scale collaborative planning: Answering high-level search queries using human computation. *AAAI*.

Manikonda, L.; Chakraborti, T.; De, S.; Talamadupula, K.; and Kambhampati, S. 2014. AI-MIX: How a planner can help guide humans towards a better crowdsourced plan. In *IAAI*.

McDermott, D.; Knoblock, C.; Veloso, M.; Weld, S.; and Wilkins, D. 1998. PDDL–the Planning Domain Definition Language: Version 1.2. *Yale Center for Computational Vision and Control, Tech. Rep. CVC TR-98-003/DCS TR-1165*.

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.

Ramírez, M., and Geffner, H. 2010. Probabilistic plan recognition using off-the-shelf classical planners. In *AAAI*.

Talamadupula, K.; Benton, J.; Kambhampati, S.; Schermerhorn, P.; and Scheutz, M. 2010. Planning for human-robot teaming in open worlds. *ACM Transactions on Intelligent Systems and Technology (TIST)*.

Talamadupula, K.; Kambhampati, S.; Hu, Y.; Nguyen, T. A.; and Zhuo, H. H. 2013. Herding the crowd: Automated planning for crowdsourced planning. In *HCOMP (Works in Progress / Demos)*.

Toutanova, K.; Klein, D.; Manning, C. D.; and Singer, Y. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proc. of HLT-NAACL*, 252–259.

Yang, Q.; Wu, K.; and Jiang, Y. 2007. Learning action models from plan examples using weighted MAX-SAT. *Artificial Intelligence Journal*.

Zhang, H.; Law, E.; Miller, R.; Gajos, K.; Parkes, D.; and Horvitz, E. 2012. Human computation tasks with global constraints. In *CHI*, 217–226.

Zhang, H.; Andre, P.; Chilton, L.; Kim, J.; Dow, S. P.; Miller, R. C.; MacKay, W.; and Beaudouin-Lafon, M. 2013. Cobi: Communitysourcing large-scale conference scheduling. In *CHI Interactivity 2013*.