Human Computation and Crowdsourcing: Works in Progress and Demonstration Abstracts
AAAI Technical Report CR-13-01

Herding the Crowd: Automated Planning for Crowdsourced Planning*

**Kartik Talamadupula, Subbarao Kambhampati,
Yuheng Hu, Tuan Nguyen, Hankz Hankui Zhuo**
Dept. of Computer Science and Engineering
Arizona State University, Tempe AZ 85287 USA
{*krt, rao, yuheng, natuan*} @ *asu.edu*, *zhuohank @ mail.sysu.edu.cn*

1 Introduction

An emerging application of human computation concerns that all too human of activities – planning. At first glance, crowdsourced planning applications appear to have very little to do with existing automated planning methods, since they seem to depend solely on human planners. However, a deeper look at these applications shows that most of them use primitive automated components in order to enforce checks and constraints which are traditionally not the strong-suit of human workers – herding the proverbial sheep, in a manner of speaking. More importantly, experiments show that even these primitive automated components go a long way towards improving plan quality (Zhang et al. 2012), for little to no investment in terms of cost and time. While these primitive components can and should be replaced by more sophisticated automated planning techniques, adapting such technology presents several challenges. We present a general architecture that foregrounds the potential roles of an automated planner in crowd-planning, and discuss the challenges in realizing them.

2 Planning for Crowdsourced Planning

The crowdsourced planning problem involves returning a plan as a solution to a task, usually specified by a user or *requester*. The requester provides a high-level description of the task – most often in natural language – which is then forwarded to the crowd, or *workers*. The workers can perform various roles, including breaking down the high-level task description into more formal and achievable sub-goals (Law and Zhang 2011), adding actions into the plan that support those sub-goals (Zhang et al. 2012), or propose further refinements to the task. These refinements can in turn be approved or rejected by the requester, if they choose to remain part of the loop. 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 plan satisfactory to the requester is found.

*Extended version available at http://arxiv.org/abs/1307.7720.

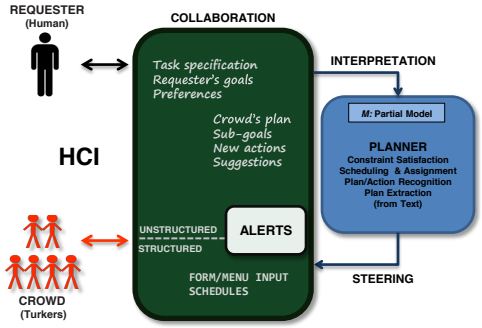

Figure 1: *An architecture that foregrounds planning challenges in crowdsourced planning systems.*

Automated planners have long tolerated humans in their decision cycle (cf. mixed initiative planning). The role of planning in human computation systems, however, differs in significant ways. In contrast to a traditional planner that expects a formal specification of the planning problem and a complete domain model, automated planners in crowd-planning scenarios have to contend with partial and evolving specifications of the planning problem, as well as partial models of the planning domain (cf. model-lite planning (Kambhampati 2007)). Further, the main role of the planner is not to "solve" the problem, but rather to effectively steer the crowd, with their own partial understanding of the problem and domain, towards more effective solutions. In the following, we focus on two critical challenges.

3 Challenges for Automated Planning

As shown in Figure 1, a planner (automated system) would interact with the rest of the system to facilitate one of two main tasks: (i) *interpretation*; and (ii) *steering*. These tasks define the planner's role in the entire process. Interpretation is required for the planner to inform itself about what the crowd *is currently* doing; steering is required to tell the crowd what they *should* be doing. The planner also needs to enable efficient collaboration amongst the crowd.

3.1 Interpretation of the Crowd's Evolving Plan

The planner must interpret the information that comes from the requester and from the crowd workers in order to act on

it. There are two ways in which the planner can ensure that it is able to understand that information:

Force Structure: The system can enforce a predetermined structure on the input from both the requester, and the crowd; an example of this is manifested in the Mobi (Zhang et al. 2012) system. The structure can be seen as part of the planner's model, since the planner has a clear idea about the kind of information that can be expected through specific channels. For example, in a travel-planning application, the requester can be asked to make selections from a fixed form, instead of being allowed to enter free-form text.

Extract Structure: The planner can also *extract* structure from text that is produced by the workers while collaborating (Kim, Chacha, and Shah 2013), in order to determine the current state of the crowd-planning process. The specific extraction method used may vary from methods that extract from plain text and impose structure (Ling and Weld 2010), to plan extraction which tries to obtain a structured plan from unstructured text (Addis and Borrajo 2011). Although this problem has connections to plan recognition (Zhuo, Yang, and Kambhampati 2012), it is significantly harder as plans need to be recognized not from actions, but rather textual descriptions.

3.2 Steering the Crowd's Plan

After determining what is going on in the planning process, the planner must then earn its due by offering helpful suggestions, alerts and perhaps even its own plan. There are various kinds of possible feedback:

Problem Identification: Even with a very simple model, the planner can be used as a basic automated constraint and arithmetic checker. For example, if the only thing known as part of the planner's model is the objective of minimizing the number of questions that the crowd has to answer (Lotosh, Milo, and Novgorodov 2013), then the automated planner is restricted to just enforcing that constraint.

Constructive Critiques: Once the planner has some knowledge about the workers' proposed plan, it can try to actively help the creation and refinement of that plan by offering suggestions. These suggestions can vary depending on the depth of the planner's model – from simple notifications of constraint violations, as outlined previously; through suggestions on the order of actions in the plan and even what actions must be present; and on to highlighting certain plans or actions over others because they satisfy the requester's stated preferences or constraints better.

Model Evolution: Given that the crowd's plan may be realized as free-form text which could contain actions and keywords that are not present in the planner's model, the planner may also ask the crowd workers to: (i) "explain" the role of those actions with respect to subsequent actions; or (ii) confirm the applicability of those unknown actions with respect to preceding actions. Such alerts can eventually also help the planner update its model.

Preference Handling & Elicitation: Approaches can range from already implemented methods, like generating a diverse set of plans for the crowd or the requester to pick from (Nguyen et al. 2012) (implicit preference elicitation), to making the crowd explicitly enumerate the preferences that the requester might hold.

Scheduling & Optimization: In certain cases, the actions produced by the crowd still need to be scheduled to create a plan. The automated system can be used to perform this scheduling – when the model is detailed enough, the system can even be used to perform optimization to produce the best plan from the suggestions mooted by the crowd.

4 Summary & Conclusion

In this paper, we took a first step towards investigating the opportunities and challenges for automated planning technology in crowdsourced planning scenarios. We identified two important challenges in adapting such technology: *interpreting* the requester inputs as well as human worker plans; and *steering* the plan generation process in the presence of incompleteness of requester preferences as well as the planner domain model. We discussed several ways in which these challenges can be tackled, and also characterized the specific (if primitive) choices made by existing crowdsourced planning systems in handling these. In the extended version (available at http://arxiv.org/abs/1307.7720), we show that existing crowd-planning systems can be understood and analyzed in terms of how they solve the two challenges. We hope that this work will spur directed research on the challenges identified.

Acknowledgments: Special thanks to Mi Jun and Jing Shan for their input. This research is supported by the ONR grants N000140910032 and N00014-13-1-0176, the NSF grant IIS-0905672, and a Google research award.

References

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

Kambhampati, S. 2007. Model-lite planning for the web age masses: The challenges of planning with incomplete and evolving domain theories. *AAAI*.

Kim, B.; Chacha, C.; and Shah, J. 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*.

Ling, X., and Weld, D. S. 2010. Temporal information extraction. In *AAAI*.

Lotosh, I.; Milo, T.; and Novgorodov, S. 2013. CrowdPlanr: Planning Made Easy with Crowd. In *Data Engineering (ICDE)*. IEEE.

Nguyen, T. A.; Do, M.; Gerevini, A. E.; Serina, I.; Srivastava, B.; and Kambhampati, S. 2012. Generating diverse plans to handle unknown and partially known user preferences. *Artificial Intelligence*.

Zhang, H.; Law, E.; Miller, R.; Gajos, K.; Parkes, D.; and Horvitz, E. 2012. Human Computation Tasks with Global Constraints. In *Proceedings of CHI*, 217–226. ACM.

Zhuo, H. H.; Yang, Q.; and Kambhampati, S. 2012. Action-model based multi-agent plan recognition. In *Advances in Neural Information Processing Systems 25*, 377–385.