Task Allocation for Crowdsourcing using AI Planning

Leticia Machado,
Rafael Prikladnicki,
Felipe Meneguzzi
Computer Science School, PUCRS
Porto Alegre, Brazil
leticia.smachado@gmail.com,
[rafaelp, felipe.meneguzzi]@pucrs.br

Cleidson R. B. de Souza
Vale Institute of Technology
UFPA
Belém, Brazil
cleidson.desouza@acm.org

Erran Carmel
Kogod School of Business
American University
Washington DC, USA
carmel@american.edu

ABSTRACT
Crowdsourcing is a relatively new phenomenon in computer science and software engineering. In crowdsourcing a task is delivered to a crowd of participants who will work on this task. Task allocation is then an important aspect in the context of crowdsourcing. If done properly, it delivers successful results based on the answers provided by the crowd. However, task allocation in crowdsourcing is not a trivial problem. Factors like a task's requirements, the knowledge required for its resolution, and the size and heterogeneity of the participants in the crowd all impact task allocation, and therefore, the expected quality of the task results. In this case, the execution of actions from a plan, which assist the dynamic tasks' allocation in crowdsourcing systems, become relevant as an alternative solution. This paper formalizes task allocation in crowdsourcing scenarios as an artificial intelligence planning problem. Our results suggest that task allocation has several challenges when it is observed in distributed, undefined and dynamic environments, like in crowdsourcing scenarios. Our goal is to evaluate if automated planning is appropriate for providing a plan to match skills of crowd workers for the right tasks in software engineering projects. Preliminary results are presented in this paper.

Categories and Subject Descriptors
D.2.m [Software Engineering]: Miscellaneous

General Terms
Measurement, Design, Experimentation.

Keywords
Keywords – crowdsourcing, automated planning, task allocation, software engineering

CSI-SE'16, May 16 2016, Austin, TX, USA
 ISBN 978-1-4503-4158-5/16/05 $15.00
DOI: http://dx.doi.org/10.1145/2897659.2897666

1. INTRODUCTION
Crowdsourcing, or simply CS, is an approach increasingly used to group of people, to be now performed by a large and indefinite labor force that is beyond the boundaries of a company.

Broadly speaking, a CS scenario involves a *requester* (i), which is an agent interested in the resolution of a particular *task* (ii); a *platform* (iii) which is technical mechanism used to distribute the task to the *crowd* (iv) and to collect the crowd's *answers* (v); each crowd member has a particular *background* (vi) (qualification, experience, etc.), and finally, as mentioned before, crowd participants are paid a *reward* (vii) for the resolution of a task. However, the positive effects of CS are not straightforward. One needs to face several challenges including efficiency and proper task allocation, i.e., to distribute a task to available and skilled crowd members requires appropriate mechanisms for the treatment of such situation in a scalable context. Routine control and coordination tasks in a CS system and the management of existing relations between requesters and crowd workers pose additional risks and challenges to the task of identifying an appropriate set of crowd workers who will perform a certain task. In short, crowdsourcing is a complex process that requires information that is sensitive to the context, such as task requirements, crowd members' background, expected task duration, target audience or the type of reward. All these aspects are essential to the success of CS initiatives and should be carefully set by whoever is requesting the task to be performed.

Meanwhile, there has been growing interest in applying Artificial Intelligence (AI) techniques in traditional systems to solve common problems. Planning approaches, for instance, are already widely used in specific areas, such as space missions and robotics. We believe that other areas can also benefit from the application of AI Planning techniques [5], [9].

In order to reduce these challenges and increase the efficiency and the quality of the provided solution for crowdsourcing, we present the use of AI Classical Planning technique to assist in the logical formalization of tasks allocation and recommendation in the three main elements of the CS setting: requester, platform and crowd.

This work is organized in six sections including this introductory section. In Section 2 we present the background of this research. Section 3 presents the research method and study setting. Section 4 analyses the results and discusses how automated planning could assist on the allocation tasks in Crowdsourcing settings. Section 5 compares this paper to related work. Finally, the conclusions and the future work are presented in Section 6.

2. BACKGROUND

A. Crowdsourcing Process
The main elements in the CS process are: the requester, the CS platform and the crowd. The requester is an agent that submits a task

and validates the solutions proposed by the crowd. Requesters also assign a financial value to be paid for completed and selected task for crowd workers. The CS platform coordinates reception and distribution of tasks among the crowd workers. Finally, the third element is represented by the crowd involved in the solution to the required tasks [3].

Current literature on CS presents several challenges that are addressed to the development of CS systems, namely how to implement the communication interface between the consumer and the supplier and how to structure its components with respect to the flow driving a task. To emphasize the need of a semantic standard, the work of Hetmank [4] presents some of the main challenges that CS platforms face: the tasks allocated to an undefined large group of corporate internal and external workers, tasks requirements and users' qualifications.

To solve large and complex human computation tasks in the CS environment is a prerequisite to have a group formation or self-organization of people with either similar or diverse, cross-functional skills or background. Unfortunately, most existing crowdsourcing systems fall short of facilitating the flexible, dynamic, and proactive assembly of globally distributed teams [2]. To propose the right task to the right person at the right time is one of the key challenges to the success of a crowdsourcing initiative.

B. Automated Planning

Artificial Intelligence planning, also known as classical planning comprises a set of techniques that aims to solve search problems by making certain assumptions about the problem domain. These assumptions allow problems to be defined using a formal representation of a domain of interest encoded in a planning language, such as PDDL [6]. PDDL, in its most basic form, allows the description of a planning problem in terms of an initial state, a goal state and a set of operator templates that can be instantiated to generate a *plan* that is a sequence of operators that transform the initial state into the goal state.

Classical planning has been used to solve not only AI problems such as robot planning [7], but also commercially relevant problems, such as business process management [8]. Consequently, in this paper, we propose the use of planning technology to plan for task allocation in crowd-sourced problem solving. Our approach encodes dynamic tasks assignment to crowd workers as planning problems. The resulting planning problem aims to reach an optimal (or nearly optimal) task assignment to workers according to their individual expertise, minimum reward and availability.

3. RESEARCH METHOD

By simulating crowdsourcing tasks and crowd, we aim to find out answers to the following key research questions:

(R1) Can a classical planning language be expressive enough to deal with the allocation tasks problem in CS scenarios?

(R2) Is it possible to generate plans to find and allocate people who fit the established criteria for the CS tasks?

(R3) Is it possible to construct plans to reduce the time of the open tasks and optimize its allocation to crowdsourcing workers?

C. Implementing the Automated Planning Algorithm for CS

In order to define and relate the main components and constraints of the CS task assignments problem, we use PDDL, a planning language for AI planning systems. PDDL is built on the top of a model in which first a domain description is formally defined, and then problems that represent instances of situations where the search should be performed are defined. In the case of a CS task allocation problem, the domain must contain the definition of objects, predicates and actions related to tasks and crowd, as well as all features, criterias and constraints that are relevant to the accomplishment of a plan.

Description of the Domain Objects and Predicates: Suppose we have a situation in which a task is released by a requester and needs to be resolved by the crowd. The crowd must meet the specifications to solve the task and, similarly, the task proposal presents prerequisites for its resolution. In this situation, the agent is the platform, the initial state is the outstanding tasks on the platform and the goal is the task allocation for workers who meet the task requirements.

To improve the tasks assign plan a group of the elements deliberate were defined to crowdsourcing problem.

TABLE I. CORE ELEMENTS

Elements	Description
task (t)	Represents a atomic task to be performed
person (p)	Represents the crowd workers registered in the crowdsourcing platforms
skill (s)	Specifies a specific skill and/or piece of knowledge
duration (d)	Specifies a defined period of time
reward (r)	indicates a precise monetary value

We model the following predicates for tasks, person and plan according Table II, Table III and Table IV.

TABLE III. PERSON PREDICATES

Elements	Description
has_skill	indicates that a person has an specific skill
avaiable_for	describes the expertise or competencies of a person
accepts_reward	indicates that a person accepts a certain reward for completing a task

TABLE II. TASK PREDICATES

Elements	Description
task_open	Indicates that a give task is opened to be developed
task_done	Indicates that a give task is completed
pays_reward	Inform an specific reward that the task will pay
demand_skill	Specifies the skill required to perform the task
has_duration	Defines the period of time estimated to complete the task

TABLE-IV. PLAN PREDICATES

Elements	Description
matches	Indicates that, all other constraints of skill, duration and reward being satisfied, a person matches to a task
task_assigned	Sets that a task is actually assigned to a person
reward_received	Inform that a reward is given to a person after the task accomplishment

Description of the Domain Actions: Some actions were defined in the algorithm aiming to generate a intelligent plan for task allocation.

- **find** - Attempt to match a person to a given task considering all constraints of skill, reward and duration
- **alocate -** Consolidate the assignment of a task to a given person if the task is open
- **resolve** - Mark the task as resolved and pay the reward

Description of the Problem

Fig. 1 shows the task specification context established in the allocation conditions represented by our research for a person and a task such as: skill, reward and duration. As described in the problem domain, it is possible analyze as a CS platform could behave to find a task and based on its structure definition, the platform could allocate which groups of people most suited to the development and solution of the task.

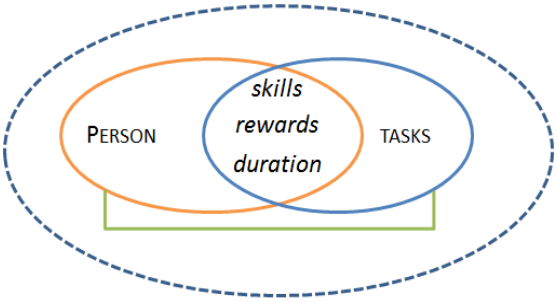

Figure 1. Intersection relation of the CS elements

It is import to understand objects' intersection of the Person and Task with the precondition, skill, reward and duration This interesection set a relationship between:

- contribuition which a worker hope receive `(accepts_reward ?p ?r)`and the amount of the money that the task will reward `(pays_reward ?t ?r)`.
- skill that a person have `(has_skill ?p ?s)` and the certain skill that the task demand `(demand_skill ?t ?s)`
- how long the task takes to complete `(has_duration ?t ?d)` and how long the work has available to resolve the task `(available_for ?p ?d)`

For the execution of the plans, problem files have been created to explore the different situations and arrangements between objects *person and task,* within state space representation of this work.

The problem file that incorporates the set of all the necessary objects for combinations suggested, it stars of the initial state where a set "t" tasks to be allocated to workers who meet the set of the skills required to solve the task. In addition it must satisfy the criterion of the reward and duration where the reward offered by the task should motivate the worker to accept the task, and the same worker needs to have time to accomplish the task.

The goal and the final state of the planner is to assign the largest number of open tasks on the CS platform for a list of people who meet the allocation criteria.

Generated Plans

In this session we describe two scenarios with different configuration for the predicates connected to person and task. The goal is to change the variables of skill (S), reward (R) and duration (D) in order to represent some normal and edge cases in the task allocation domain. Each configuration has generated different results, showing both the strengths and weakness of using the classical planning approach.

Scenario 1:

This scenario is aimed to describe a simple successful path, where each task that demands some skill, reward and time has at least one matching person with that skill, accepting that reward, and available for that time. In the crowdsourcing context, this scenario is represented by a configuration where there are enough people in the crowd to match all tasks requirements, with no need of adjusting any of the variables of the system.

P1 has S1, S2, S3, accept R1 and is available D1, D2. D3
P2 has S2, S3, accept R1, R2 and is available D1, D2. D3
P3 has S3, accept R1, R2, R3 and is available D1, D2. D3

T1 requires S1, pays R1 and takes D1
T2 requires S2, pays R2 and takes D2
Tn requires Sn, pays Rn and takes Dn

The generate plan chooses, for each task, a person with the matching variables.

Scenario 2:

This scenario defines a situation where, for one single task, there is not a matching person in the crowd. More specifically the crowd matches the skill required for this task, as well as the reward that it pays, but there is no available person for the duration of the task.

P1 has S1, accepts R1 and is available for D1
P2 has S2, accepts R2 and is available for D2
Pn has Sn, accepts Rn but is **not** available for **Dn** (and no other person in the crowd is available).

T1 requires S1, pays R1 and takes D1
T2 requires S2, pays R2 and takes D2
Tn requires Sn, pays Rn and takes Dn

As a result, no plan was generated for this scenario, since the planner, that had the goal of allocating all tasks, couldn't find a person matching one of tasks duration. When analyzing each task set of requirements, if there is no person in the crowd that matches this set, the algorithm will fail in finding a plan. No partial plans – plans that ignore one or more tasks that are not matched by a person in the crowd – are given as a solution.

In this case, the feedback that the system gives is that an adjustment of the variables is needed in order to have a successful plan. More specifically, the duration of the task (Tn) must be relaxed to a value that is possible to match with any of the people in the crowd.

Scenario 3:

This scenario aims to adjust the duration of a task that was previously unmatched to any person in the crowd (according to Scenario 2). The duration was set to a value that more people in the crowd were available for, and, as the result, a plan was found for the scenario.

Allowing changing variables in the scenario configuration and testing different strategies is one of the features of the current algorithm. Someone may start a scenario with the lowest cost, faster duration and high level skills and try finding a plan. While no plan is found the values of the variables can be gradually adjusted in order to get to a plan that is effective and low cost.

Scenario 4

This scenario configures tasks that require more than one skill for its resolution.

T1 requires S1 and S2, pays R1 and demands D1
T2 requires S2 and S3 pays R2 and demands D2
Tn requires Sn1, Sn2 and Sn3, pays Rn and demands Dn

A plan was generated that found a set of people that could be allocated to the skill demanding tasks. Since there were only a few people in the crowd that had the necessary skills, more than one task was given to them.

4. REVISITING THE RESEARCH QUESTIONS

R1 - Can the classical planning language be expressive to deal with the allocation tasks problem in CS scenarios?

Even though in a preliminary way, it was possible to evaluate the application of techniques of classical planning to the formalization of problems within the context of crowdsourcing requirements. The use of planners allowed the evaluation of different scenarios based on the attributes that were configured to represent the CS environment in our model.

R2 - Is it possible to generate plans to find and allocate people who fit the established criteria for the CS tasks?

The algorithm generates a plan whenever the requirements for each task in the scenario are matched to at least one person in the crowd. In the case were there are one or more tasks that don't have its requirements matched, the algorithm will fail to output a plan.

This may suggest that the system will always look for people that meet optimally or sub optimal preconditions of the tasks seeking ensure quality in the delivery of solutions through the variable configuration skill.

R3 - Is it possible to construct plans to reduce time of the open tasks and optimize its allocation to crowdsourced works?

In the model developed for this paper, considering only the classical planning approach, and not using any cost function to evaluate the optimization of the attributes, is not possible to construct plans that will optimize in any of the variables.

The main result of the current algorithm is to indicate that a plan is found, whether is optimal or not. This allows the testing of several configurations and the increasing or decreasing of any of the variables.

Meets the objective of finding a suitable set of people that matches the tasks according to the configuration of the context.

R4 - Can the outcome plans improve the quality of the providers solutions?

In order to build different scenarios and setting the variables we can have tradeoff a tradeoff of cost versus quality in some cases the requesters will have to choose between having a plan that meets delivery of quality aspects and in such cases the reward needs to be high to motivate people with desired qualifications to accept the task execution. And in cases where the lower cost for task execution is priority, the planner will check for people with less reward to allocate tasks in detriment to quality.

The categories of knowledge created from the set of skills mapped the workers participating in the CS platform can facilitate the combination of right people for the right tasks.

The described formalism for the problem is based on an environment completely observable, deterministic, static, finite and discrete. However, despite these limitations, this kind of planning was able to offer a certain challenge for the context of this work, for, event with small sized problems, a big amount of states are created and several restrictions in the representation of the domain are needed.

Questions like diversity of knowledge in the crowd and in the resolutions of tasks may be compromised, because, considering only the set of skills defined by the worker in the platform profile, the planner is restricted to search only people that are declared in the task and in the worker.

In this case, if it was possible to obtain other information from the CS platform, for example, the worker reputation, the planner could consider experience and quality attributes of the worker and, also, the tasks could be registered to match to workers with certain reputation and quality. Yet the usefulness and potential of AI techniques in Crowdsourcing setting has only begun to be explored. Crowd-sourced allocation and scheduling problems are discussed in terms of architecture and process in the literature as presented in the section 5.

5. RELATED WORK

The application of automated planning in crowdsourcing planning problems has some interesting related work. In order to identify challenges in adapting automated planning technology in crowdsourcing planning scenarios, Talamadupula et al. [9] show a general architecture for human computation (crowdsourced) systems with a view of the roles of an automated planner. There are two key differences between the two areas of CS. First, is the random versus guided behavior in the assignment of workers in designed in the CS system. Second, is that often performance results in open CS can be poor and incomplete. In contrast, the coordination algorithms current used in guided CS is herding the crowd towards more effective solutions

One way to evaluate the use AI methods to coordinate a user crowd towards achieving specific collective performance goals in a crowdsourcing setting is presented by Lykourentzou et al. [5] in the new area called guided crowdsourcing. They propose the process of engineering a guided crowdsourcing that was used in a test case in corporate crowdsourcing. They implemented the algorithm which uses resource scheduling to dynamically assign micro-tasks to workers. The evaluation of this algorithm in two real scenarios demonstrated in terms of quality, cost and timeliness by using AI-based methods of crowdsourcing systems.

In Mao et al. [10], the authors showed a recommender system to recommend developers in the SW CS environment in order to help the platform find reliable and suitable developers, who may be interested in registration for promoting the task participation in TopCoder platform.

The three works discussed in this section contributed with interesting insights for this paper. Our work differs from them because we propose the use of Artificial Intelligence planning to assist in the evaluation of different task allocation scenarios based on the set of variables that have been configured to represent the CS environment.

6. CONCLUSIONS AND FUTURE WORK

As show in the previous sections, our goal in this paper is to evaluate if PDDL language – a logical formalism, is appropriate for providing a plan to match skills of crowd workers for the right tasks in software engineering projects.

As a contribution, the modeling and the preliminary results obtained for this study indicate that the automated planning has interesting mechanisms for generating plans that can contribute significantly to description and standardization of other prerequisites that define a task and a user on CS applications. Moreover, it was possible to show that the algorithm allows changing variables in the scenario configuration and testing different strategies in the tasks allocation.

The modeling and results obtained for this study indicate that PDDL has interesting mechanisms for generating plans that can contribute significantly to the description and standardization of other prerequisites that define a task and a user on CS applications. The adoption of classical planning techniques can improve CS task allocation from a deliberate crowd as pull model to a push model or guided crowdsourcing.

As future work, we aim to focus on using the cost function to optimize the solutions generated by automated planning for allocation crowdsourcing. One design solution could be that the requesters specify the minimal cost and the minimal skill required for their work. Therefore, the more the workers is willing to considered for assignment, and, as it is sensible to assume, more suitable workers could be found.

Also, we foresee that a solution should be hybrid, i.e., support both allocation and scheduled approaches, so we plan to investigate the possible hybrid approaches and correspondent integration issues in this field.

ACKNOWLEDGMENT

This work is partially sponsored by the Brazilian Law 8.248/91. This research is also partially funded by CNPq (406692/2013-0 and 301136/2014-9). Rafael Prikladnicki is a CNPq researcher (312127/2015-4). Felipe Meneguzzi thanks CNPq for support within process numbers 306864/2013-4 under the PQ fellowship and 482156/2013-9 under the Universal project programs.

REFERENCES

[1] Zhao, Y. and Zhu, Q. “Evaluation on crowdsourcing research: Current status and future direction”. Springer Science+Business Media, LLC, 2012.

[2] Vukovic, M. “Crowdsourcing for enterprises”. In Proceedings of the Congress on Services, pp. 686–692. IEEE Computer Society, 2009.

[3] Machado, L., Pereira, G., Prikladnicki, R., Carmel, E., & de Souza, C. R. “Crowdsourcing in the Brazilian IT industry: what we know and what we don't know”. In Proceedings of the 1st International Workshop on Crowd-based Software Development Methods and Technologies (pp. 7-12). ACM, November 2014.

[4] Hetmank, L. “Towards a Semantic Standard for Enterprise Crowdsourcing – A Scenario-based Evaluation of a Conceptual Prototype”. In 21st European Conference on In-formation Systems (ECIS). Utrecht, 2013.

[5] Lykourentzou, I., Vergados, D. J., Papadaki, K., & Naudet, Y. “Guided crowdsourcing for collective work coordination in corporate environments”. In Computational Collective Intelligence. Technologies and Applications (pp. 90-99). Springer Berlin Heidelberg, 2013.

[6] E. Gerevini, P.Haslum, D. Long, A. Saett amd Y. Dimopoulos, “Deterministic planning in the fifth international planning competition: PDDL3 and experimental evaluation of the planners”, Artificial Intelligence, Volume 173, Issues 5–6, April 2009, Pages 619-668.

[7] S.Srivastava, E.Fang, L.R. R. Chitnis, S.Russell, and P. Abbeel. “Combined task and motion planning through an extensible planner-independent interface layer” In Proceedings of the IEEE Conference on Robotics and Automation (ICRA), 2014, pages 639—646.

[8] J.Hoffmann, I.Weber and F. M. Kraft "SAP Speaks PDDL: Exploiting a Software-Engineering Model for Planning in Business Process Management", Journal of Artificial Intelligence Research, Volume 44, July, 2012, pages 587-632.

[9] Talamadupula, K., et al. "Herding the crowd: Automated planning for crowdsourced planning." First AAAI Conference on Human Computation and Crowdsourcing. 2013.

[10] K. Mao et al. "Developer Recommendation for Crowdsourced Software Development Tasks." *Service-Oriented System Engineering (SOSE), 2015 IEEE Symposium on*. IEEE, 2015.