Predicting QoS in Scheduled Crowdsourcing

Roman Khazankin, Daniel Schall, and Schahram Dustdar

Distributed Systems Group, Vienna University of Technology,
Argentinierstrasse 8/184-1, A-1040 Vienna, Austria
lastname@infosys.tuwien.ac.at
http://www.infosys.tuwien.ac.at

Abstract. Crowdsourcing has emerged as a new paradigm for outsourcing simple for humans yet hard to automate tasks to an undefined network of people. Crowdsourcing platforms like Amazon Mechanical Turk provide scalability and flexibility for customers that need to get manifold similar independent jobs done. However, such platforms do not provide certain guarantees for their services regarding the expected job quality and the time of processing, although such guarantees are advantageous from the perspective of Business Process Management. In this paper, we consider an alternative architecture of a crowdsourcing platform, where the workers are assigned to tasks by the platform according to their availability and skills. We propose the technique for estimating accomplishable guarantees and negotiating Service Level Agreements in such an environment.

Keywords: Crowdsourcing, Scheduling, QoS, SLA, Negotiation.

1 Introduction

Enterprises seek to automate all sorts of their front and back office operations to cut costs and eliminate human factors. However, humans inevitably remain as key elements of many business processes. It is not only creative, management, and communication activities that are hard to reproduce with technology. Other tasks that require basic human skills, such as image recognition or categorization, as well as specific skills, such as text translation or usability testing, are also difficult to fulfill with software only. Crowdsourcing remains a prominent paradigm for solving such tasks and providing human workforce on demand. Recent efforts demonstrate the successful adoption of crowdsourcing techniques at an ever-increasing rate, and the amounts of both platforms and workers in such platforms are expected to grow rapidly[5].

Crowdsourcing systems are diverse in their architecture, target problems, and user interaction styles [3]. We focus on platforms that deal with tasks consisting of manifold similar independent jobs provided by consumers, such as translation, reviewing, voting, tagging, content creation, image recognition, data quality improvement, and so on. Currently, such platforms have a market-like operation chain where the tasks received from customers are announced at the portal and the workers choose among the assorted mass of tasks those they like to process.

J. Ralyté et al. (Eds.): CAiSE 2012, LNCS 7328, pp. 460–472, 2012.

Examples of such systems include Amazon Mechanical Turk[1](AMT), CrowdFlower[2] and ClowdCrowd[3].

Although the aforementioned systems are considered quite successful, we argue that purely market-like architecture lacks some features that could realize more potential of crowdsourcing platforms. As in a market-like system the job assignments are initiated by the workers themselves, it is hardly possible for the system to have an active influence upon assignments. As a result, the platform is unable to give any certain guarantees for the consumers, neither about the time of processing a task nor about the outcome quality they can expect[9]. From a Business Process Management (BPM) perspective, such guarantees can provide an additional value for crowdsourcing services. First, it can strengthen the certainty and predictability in process planning and design. Second, if such guarantees are given in form of Service Level Agreements (SLAs), it allows for crowdsourcing in QoS(Quality of Service)-sensitive business processes [1,2,6].

In our previous work [12] we presented a crowdsourcing platform model where the workers are assigned to tasks *by the platform* according to current workers' availability, skills of workers, skill requirements provided by consumers, and service level agreements with consumers (described in Sec. 2). We refer to this model as *scheduled crowdsourcing*. While providing the same flexibility, scheduled crowdsourcing comprises a number of advantages:

- **Quality.** Skills of the workers are manifold. The tasks submitted to the platform are also diverse in their requirements. One can assume that the more the worker is suitable for a task, the better the expected outcome quality is. We refer to this indicator as *suitability*. Hence, by considering the worker-task suitability, it is possible to improve the overall results by assigning tasks to most suitable workers.
- **Deadlines.** In market-like platforms task completion times span from several to thousands of hours [8]. As in scheduled crowdsourcing the assignments are controlled by the platform, tasks can be scheduled according to specified deadlines.
- **Predictions and SLAs.** Considering the short-term information about workers' availability on the one hand and tasks in progress on the other hand, the platform can predict the available workforce and, thus, estimate what can be offered or guaranteed to a consumer who wants to submit a particular task.

In this paper we focus on prediction and SLA negotiation in scheduled crowdsourcing. As mentioned above, SLAs provide an additional value for services. However, when an SLA is negotiated with a customer, the platform has to make sure that this SLA is feasible and will not endanger other agreements. Specifically, we address the following questions:

[1] `http://www.mturk.com/`
[2] `http://www.crowdflower.com/`
[3] `http://www.cloudcrowd.com/`

- *How fast a task can be done?* If too many jobs are scheduled to the same period, there could be not enough available workers to withstand the workload, so some deadlines will be broken. Thus the platform should determine the *earliest deadline* which the customer could set up for his/her task such that the timely execution of other tasks is not endangered.
- *What is the expected quality of the result?* As mentioned above, different outcome quality can be expected from different workers. Thus, if a close deadline it set, then more workers must be involved, and, therefore, the average result quality could be lower than in the late deadline case, where the smaller amount of best workers would do all the jobs. Therefore, there is a trade-off between the task deadline and the resulting average quality of the task. Estimating and explicitly presenting such a trade-off to the consumer would clarify what s/he can expect when submitting a task. Moreover, it can serve as a basis for SLAs which, as mentioned before, are crucial in a service-oriented environment. To achieve this, the platform needs to predict *quality by deadline* efficiently for multiple deadlines.

The paper introduces a technique and a base algorithm for predictions in scheduled crowdsourcing environment. The general idea of the technique is to simulate the work of the platform using the prior experience data. The precision and performance of the approach are evaluated through experiments.

The paper is organized as follows: Section 2 describes the environment and the platform model. The approach and the algorithm are presented in Sec. 3 and evaluated in Sec. 4. Section 5 discusses the related work. In Sec. 6, we shed light on some disputable aspects of our model. Section 7 concludes the paper.

2 Scheduled Crowdsourcing

This section describes a platform model that supports scheduled crowdsourcing [12]. The platform receives tasks from consumers and distributes these tasks for execution to the crowd. A task comprises manifold similar independent jobs which can be assigned to workers. When a job is done, the result is returned to the consumer, which is invited to provide a quality feedback on this result (see Fig. 1).

Before a consumer submits a task, an SLA for this task is negotiated. The SLA includes temporal and quality requirements. Figure 2 depicts the process of negotiating the SLA. At first, the consumer provides the parameters and skill requirements of the task to the platform. The parameters include the time of expected submission, the amount of jobs, and the job duration. Secondly, the platform estimates possible options regarding the processing time and the average outcome quality considering the status of the crowd and other active tasks or scheduled tasks. After that, the consumer decides, which option is the most suitable, and, finally, the agreement is established. The platform takes this agreement into consideration when negotiating other agreements and scheduling the tasks. If the consumer doesn't have the actual task contents at the moment, but is certain to provide it in the near future and knows the parameters of the task, then the SLA can be negotiated in advance of the actual submission.

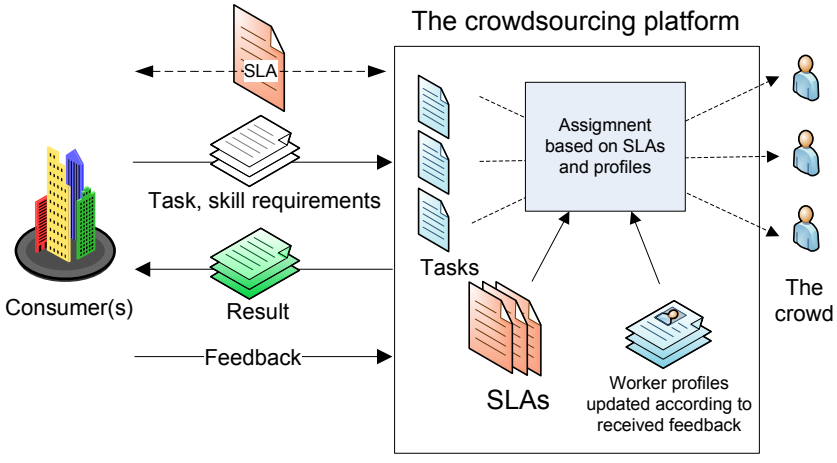


Fig. 1. Scheduled crowdsourcing

Active jobs are assigned to workers according to worker-task suitability, negotiated SLAs, and short-term availability information provided by workers. After all jobs of a task are done, the average outcome quality is calculated for the task. If it does not exceed the guaranteed quality, then the provider might incur penalties towards the consumer. If an assignment is refused by a worker despite his claim for availability, various penalty sanctions can be imposed to this worker.

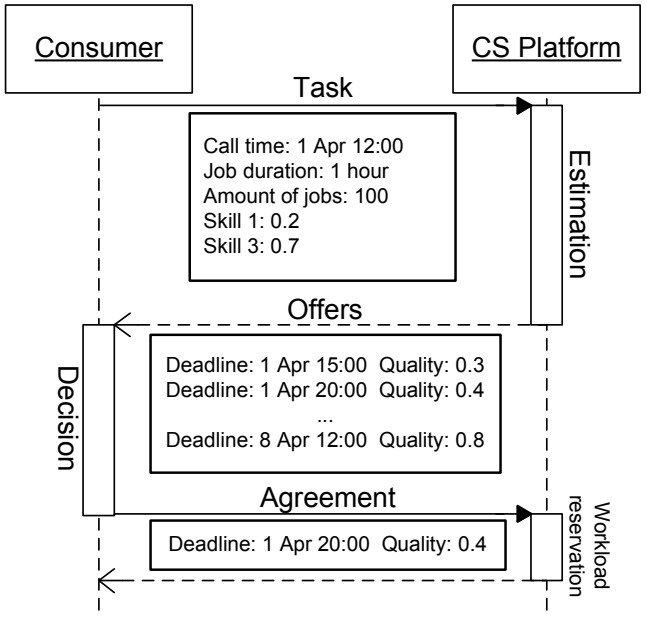


Fig. 2. SLA negotiation

The suitability is calculated as a match between required skills for the task and the skills of a worker. Skills of workers are maintained in their profiles. Initially, skill information is provided by the workers themselves. However, the profile of a worker can be modified by the platform if the expected quality (suitability) differs from the real quality that was reported by the consumer as a feedback. The maintenance of workers' skills is performed in the platform by analyzing the feedback and trying to keep the skills in the profile aligned with the real skills of the worker. A skill maintenance and update approach was considered in detail in [12], and therefore is out of this paper's focus. Moreover, the rules for calculating the suitability are independent of the scheduling and prediction logic. The suitability is represented by a single real value in $[0, 1]$ (0 - not suitable at all, 1 - perfectly suitable) which summarizes the expectations regarding the quality of the result. This assertion completely decouples the technique, which is used to calculate the suitability, from scheduling and prediction algorithms. Therefore, the architecture is compatible with other skill and suitability assessment approaches, such as in [15] or [10].

3 Prediction and Estimation

In this section we propose a mechanism for earliest deadline estimation and average quality by deadline predictions in scheduled crowdsourcing. The general idea of our approach is to actually simulate the work of the platform using the statistical data about workers' behaviour and availability, considering earlier submitted or negotiated tasks. Then it is possible to predict which and how many workers might be available for the task which is being negotiated. As was shown in [12], the greedy assignment algorithm is generally good enough for scheduling in crowdsourcing environments due to a large number of available workers. Therefore, due to the linear complexity of the algorithm, the simulation can perform near real-time.

To achieve a verisimilar simulation, the simulated environment should mimic the real conditions. From the scheduling perspective, the following characteristics are important:

- **Worker's availability.** Although it is impossible to predict whether a particular worker is available at particular time, the approximate availability of each worker can be predicted from the history and the reported short-time availability. In our implementation, the availability of each worker is generated randomly for the simulated period based on his/her recent schedule.
- **Job duration accuracy.** The time that a worker needs to finish a job can differ from the specified job duration. The reasons can be an inaccurate estimation of the job duration from the consumer, the slow speed of the worker, or the difficulty of the particular job. We discuss this issue further in Sec. 6. The accuracy for already submitted tasks can be estimated based on prior assignments of these tasks. We estimate the overall job accuracy in our simulation.

– **Suitability.** As mentioned in Sec. 2, worker-task suitability is calculated by the platform and can be modified with time. If a task of a kind is submitted to the platform for the first time, the suitability can be calculated imprecisely. However, the following task submissions of this kind (with the same skill requirements) will use refined values. We discuss this issue further in Sec. 6. In any case, the simulation can only make use of the latest calculated suitability values and assume the quality of a job equal to the suitability of the performer and the corresponding task.

A deterministic time model is used in the simulator, so the time is discrete and is represented by sequential equally long time periods. A time period can represent, e.g., an hour. Quality is described by a real number in $[0, 1]$ (0 - lowest quality, 1 - perfect quality).

The simulation period starts with the current state of the real platform. The size of the period can be either fixed (e.g., predictions for up to 10 days) or depend on the quality increment (e.g., if by prolonging the deadline by 1 day the expected quality is increased by less than 0.05, then stop the prediction). Durations of jobs are simulated according to the estimated accuracy.

At each step, the submitted (or pre-submitted) tasks are assigned to workers using the greedy scheduling algorithm (see Alg. 1), the objective of which is to maximize the overall quality, while fulfilling deadlines. The parameters of a task include the time of expected submission, amount of jobs, and job duration. The algorithms iterates over tasks in the order of times their SLAs were agreed. For each task, it tries to assign best suitable workers, so that each *task duration* an equal amount of assignments is performed. If the deadline is less than 2 *task durations* away, then all the jobs of the task are assigned. The assignment is interrupted if no more workers are available.

After conducting the assignment, the prediction algorithm is executed for the current step (see Alg. 2): workers, that are available and were not assigned by the scheduling algorithm, are examined as candidates for the negotiated task `nTask`.

Algorithm 1. Greedy scheduling algorithm.

Require: $currentTime$ current time
Require: $tasks$ active tasks
for $task \in tasks$ in the order of ascending $task.agreementTime$ **do**
  $stepsToDeadline = (task.deadline - currentTime{+}1)$ / $task.duration$ - 1
  if $stepsToDeadline > 0$ **then**
    if $(task.deadline - currentTime + 1) \,\%\, task.duration) > 0$ **then**
      $toTake = 0$
    else
      $toTake = Trunc(task.numberOfJobsToDo/stepsToDeadline)$
    end if
  else
    $toTake = task.numberOfJobsToDo$
  end if
  while $toTake > 0$ AND some workers are still available **do**
    Assign a job of $task$ to most suitable available worker
    $toTake = toTake - 1$
  end while
end for

This is performed for each job duration period of `nTask` using so-called array of average suitability of best workers (`avgSuit`). The kth element of this array represents an approximated suitability value of the kth most suitable worker for all prior simulation steps. This element contains the summed suitability and the amount of workers that were considered at this position, so the average value can be calculated at each step. The algorithm thus adds the *suitability* value and increments the *amount* for each element that corresponds to an unassigned worker (lines 3-10 of Alg. 2).

After that, if the total amount of available workers at previous steps exceeds the amount of jobs in `nTask` with certain excess, the prediction of quality is calculated for the current step. The algorithm assumes that best available workers would be evenly assigned for the task (true for the greedy scheduling algorithm). Using `avgSuit` array, it estimates the expected quality produces by most suitable available workers for all previous steps using `avgSuit` array (lines 11-21 of Alg. 2). It is assumed that a worker is late with the job with probability of 0.5 (this assumption holds if the job duration is set accurately). Eventually, the average quality which represents the prediction for *quality by deadline* for the current step, as if it was the deadline, is calculated. The *earliest deadline* is the step where it was first estimated that the number of available workers at previous steps exceeds the amount of jobs.

Algorithm 2. Prediction algorithm.

Require: $time$ current time
Require: $nTask$ negotiated task
Require: $avgSuit$ the array of average suitability of best workers
Require: $ttlAvWorkers$ total number of available workers
Require: Δ excess ratio (0.8 used)
1: **if** $(time - nTask.callTime)\ \%\ nTask.jobDuration == 0$ **and** $(time > nTask.callTime)$ **then**
2: $i = 0$
3: **for** $worker \in workers$ **in the order of descending suitability do**
4: **if** $worker$ **was not assigned and is available then**
5: $avgSuit[i].suitability+ =$ **suitability of** $worker$
6: $avgSuit[i].amount++$
7: $i++$
8: $ttlAvWorkers++$
9: **end if**
10: **end for**
11: **if** $ttlAvWorkers * 0.5 * \Delta > nTask.numberOfJobs$ **then**
12: $toTake = nTask.numberOfJobs$
13: $i = 0$
14: $q = 0$
15: **while** $toTake > 0$ **do**
16: $take = Max(1, floor(Min(avgSuit[i].amount * 0.5, toTake)))$
17: $toTake- = take$
18: $q+ = avgSuit[i].suitability * take$
19: $i++$
20: **end while**
21: **return** $\{time, q/nTask.numberOfJobs\}$
22: **end if**
23: **end if**

As the scheduling algorithm prioritizes tasks by submit time, no "collisions" are expected: on the one hand, the prediction doesn't take already reserved resources into account, on the other hand, if the task is submitted, the assessed resources will not be assumed available for the tasks submitted afterwards.

4 Experiments

To explore the potentiality of our approach, we implemented and tested the prediction mechanism in a simulated crowdsourcing environment. Simulation of such an environment is challenging due to the lack of comprehensive statistical data in this area. Although we don't rely on real data in our simulations, we tried our best to prognosticate the meaningful simulation parameters based on our knowledge and experience.

One could argue that using the simulation to predict the outcome of a simulated environment is meaningless. However, in our experiments, the prediction mechanism operates completely separately from the simulated environment. The parameters of the prediction's simulator such as availability of workers and job duration accuracy, are estimated or generated based on the simulated environment's prior activity only. Also, the duration of individual assignments, if those happen to take place in both simulators, would be different.

4.1 Setup

Customers. Tasks from customers are submitted randomly while ensuring the average crowd workload. At each simulation time period, if the *Task Limit* has not been reached yet, a new task is submitted to the system with *Task Concentration* probability. The job duration is calculated as $Min(1 + abs(\phi/2 * \sigma), \sigma + 1)$, where ϕ is a normally distributed random value with mean 0 and standard deviation 1. The deadline is assigned randomly according to *Steps To Deadline* parameter. The number of jobs is calculated so that the crowd workload is near equally distributed among the tasks, and the average workload remains close to *Intended Schedule Density*. The parameters and their values are described in Table 1.

The experiments embrace diverse tasks: some can be successfully done by most of workers, some require a special set of skills, so there are only few very suitable workers, etc. Instead of explaining the generation procedure, the suitability data from experiments for some randomly selected tasks is depicted in Fig. 3.

Crowd Workers. The crowd size in experiments was 1000 workers (except for performance tests). This size is big enough to enclose the diversity of workers, but still allows for fast simulation. In our experiments we use a *Workers Unavailability* parameter which indicates the mean ratio of unavailable workers for each period of time (values used: $0.3 - 0.6$, step 0.1). The busy periods are generated randomly, but have a continuous form which reproduces human behavior. The amount of time that takes a worker to finish the job is the *Job Duration* with injected variations. In our experiments we used values of 30-50%, which means that a job can be executed for $0.5 - 1.5$ job durations.

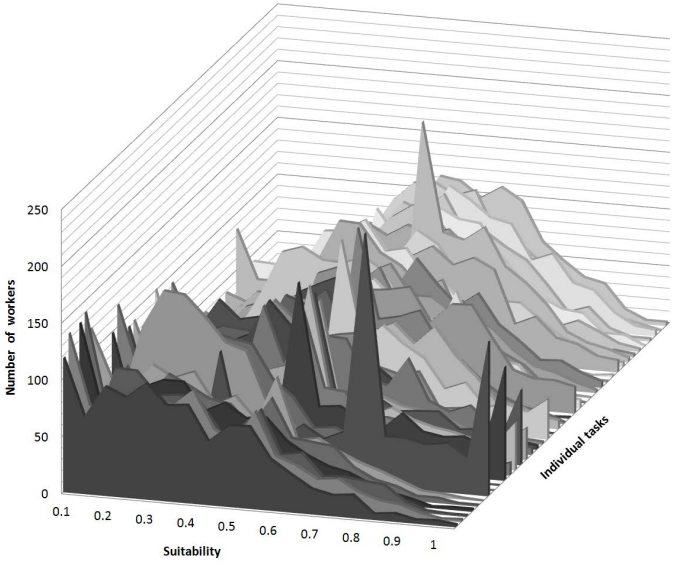


Fig. 3. Suitability of the crowd for a random set of tasks

Table 1. Task generation parameters

Name	Description	Value(s)
Tasks Limit	The total number of submitted tasks	200
Job Duration Sigma (σ)	Describes the deviation and the maximum for job durations	20
Steps To Deadline	Average maximum number of jobs of a task that a single worker can finish until the deadline.	50
Task Concentration	The probability of new task submission for each time period.	0.35
Intended Schedule Density	Target assignment ratio for each time period.	0.4 - 0.7 (step 0.1)

Such circumstances as rejecting the assignment or failing to deliver are not explicitly simulated in our setting. However, inaccurate job durations and randomized worker availability partially cover these cases. For example, a situation where one worker spends too much time for a task and another worker is temporarily unavailable after that time is similar to a situation where one worker fails to deliver and another worker gets this job re-assigned.

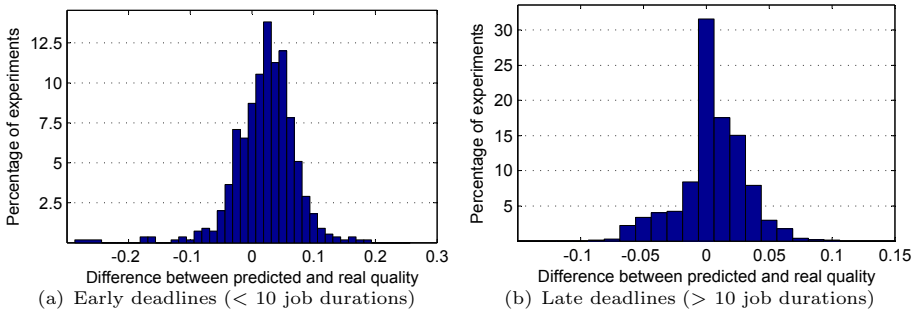
4.2 Experiment Types and Results

In the experiments we evaluated the prediction accuracy and the performance of the approach.

Prediction Accuracy. First, we made the predictions for 50 tasks in the middle of simulation for 500 time periods. It better reflects a real crowdsourcing environment, as there are both tasks being in progress and new tasks being

submitted. Then, we checked each prediction by varying the deadlines of corresponding tasks and running the simulation in the identical setting. The resulting accuracy is depicted in Fig. 4. From the total of 2041 experiment, in 98% the deviation was less than 0.1, in 85% of experiment - less than 0.05. The average deviation was approximately 0.025. Evidently, the prediction is less accurate for early deadlines, and more accurate for late dealines.

The results indicate that the algorithm can be successfully applied for negotiating the agreements. Moreover, the guaranteed values can be calculated depending on deadline remoteness. For example, the guarantee can be given as the predicted quality reduced by 0.2 in case of early deadlines, and reduced by 0.1 in case of late deadlines.



(a) Early deadlines (< 10 job durations) (b) Late deadlines (> 10 job durations)

Fig. 4. Prediction accuracy. Histograms describe the amount of experiments (in percentage of the total number of experiments performed) that produced one or another accuracy. Subfigure (a) corresponds to experiments in which the deadline was set to be less than 10 job durations of the task, SLA of which is being negotiated; Subfigure (b) corresponds to experiments in which the deadline was set to be more than 10 job durations.

Performance. We ran the prediction in the same setting while varying the size of the crowd from 1000 to 10000. The prediction overhead is depicted in Fig. 5.

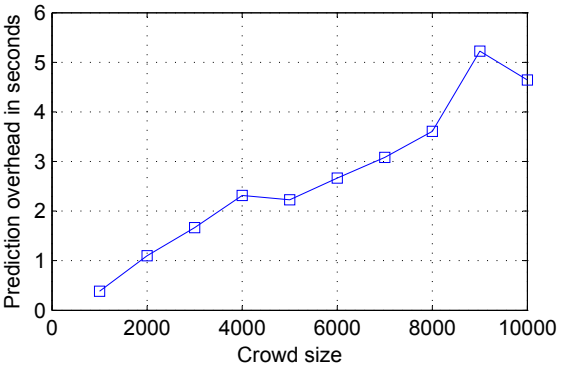


Fig. 5. Prediction performance

System parameters were Intel Core 2 Quad 2.40 GhZ with 6 GB of RAM (the algorithm is not parallelized so only one core was actually used).

The results show that the approach can be used in a near real-time setting. The overhead of several seconds would not play a huge role when negotiating the agreement and is therefore acceptable.

5 Related Work

Although the idea of QoS-enhanced crowdsourcing was discussed before [11], to the best of our knowledge, no work was devoted to deadline- and quality-centric predictions and guarantees in crowdsourcing.

In [14], semi-automatic assignment mechanism was proposed. This work assumes that SLAs are established with the workers, and Community Brokers are hold responsible for assignments in various crowd segments. However, no SLAs with crowdsourcing service consumers are considered.

Advanced market-based crowdsourcing platform which takes the suitability of workers to tasks into account was proposed in [15]. For a given task, it organizes an auction among only the most suitable workers to increase the overall quality and motivate workers to improve their skills. Again, this model doesn't provide predicting capabilities and doesn't support SLAs.

The considered scheduling problem is based on worker-task suitability, task deadlines, and workers availability with the objective of maximizing the job quality, and does not correspond to any of well-known scheduling problems [13]. The formulation can be boiled down to the problem of unrelated machines in parallel with deadlines, but the optimization objective is different from objectives related to processing time which are commonly studied in the domain.

Staff scheduling [4] designs workers' schedules which should fulfill certain requirements and cover the tasks that need to be done in a given planning horizon. Crowdsourcing's idea is the opposite: the workers define their schedule by themselves, and the platform assigns available workers to tasks in progress dynamically.

6 Discussion

In this section we discuss some disputable aspects of our approach.

The operation of the platform is influenced by several subjective characteristics provided by consumers, such as skill requirements, job duration, or feedback. However, it is of consumer's interest to specify them accurately. For example, if s/he underestimates job duration, then some deadlines might be missed, or the resulting quality can be lower than agreed. This situation can be spotted by the platform as the majority of workers would do the jobs longer than expected. Then, the input data from the consumer can be considered misleading, and the agreement can be denounced. If the consumer overestimates the job duration, then agreements are not endangered as most of workers would be faster than expected, however, this would produce underestimated predictions.

The same can be applied for other characteristics: if the platform spots that the majority of assignments do not correspond to expectations - then they were probably specified inaccurately. Moreover, guarantees are given according to the platform's sight, so the agreement can include the condition that the platform cannot be responsible for the outcome quality if the input data was inaccurate, and the consumer is responsible for the accuracy of his/her input. The consumer in turn can rely on the prior experience or submit some sample tasks to adjust these parameters.

Crowdsourcing presumes a substantial amount of registered workers, and the scheduled crowdsourcing puts even more restrictions on what the workers can do and when. The feasibility of real-world deployment of such a platform can thus be questioned. Some contrary arguments, however, are that about 20% of AMT workers consider AMT as their primary income source, and about 20% of AMT workers complete more than 200 jobs per week [7]. Considering that AMT claims that more than 500 000 workers are registered in the platform, one can conclude that there is a significant amount of people who are willing to perform jobs at the regular basis. Moreover, the payments for jobs in scheduled crowdsorcing can be bigger due to the added value of SLAs. Finally, the scheduling mechanism can be upgraded to take account of workers' preferences, while keeping the assignments compliant to the established agreements.

In this paper we don't discuss the cost of the work and payments in detail. Although it is an important factor, in our vision, it can be seamlessly integrated into the platform. One design solution could be that the workers specify the minimal cost for their work and the consumers would pay as much as they want as in a traditional crowdsourcing platform. Therefore, the more the customer is willing to pay, the more workers would be considered for assignment, and, as it is sensible to assume, more suitable workers could be found. However, such a design will not change the basics and the algorithms of our platform substantially. Thus, for the sake of simplicity, in this work we assumed that all the jobs cost correspondingly to their specified duration.

7 Conclusion

A technique and a base algorithm for predictions in scheduled crowdsourcing environment are proposed in the paper. The potentiality of the approach is evaluated thorough experiments which show that such an environment is predictive in spite of its inherent uncertainty. The proposed base algorithm can be considered rather precise (average quality deviation 0.025), and, therefore, can be applied for negotiating the agreements. The experiments show that the prediction accuracy depends on deadline remoteness, the guaranteed values therefore can be adjusted accordingly.

These results show that crowdsourcing platforms can be organized to provide quality guarantees for the consumers. They can strengthen the certainty and predictability in process planning and design, and enable Service Level Agreements with customers. From a Business Process Management perspective, such guarantees provide an additional value, thus promoting more advantageous crowdsourcing services.

In our future work, we will focus on pricing of services for scheduled crowdsourcing. Also, we foresee that a practically-applicable solution should be hybrid, i.e., support both market-like and scheduled approaches, so we plan to investigate the possible hybrid architectures and correspondent integration issues in this field.

Acknowledgement. This work was supported by the Vienna Science and Technology Fund (WWTF), project ICT08-032.

References

1. Adams, C.: Managing crowdsourcing assignments. Cutter IT Journal 24(6), 6–11 (2011)
2. Adams, C., Ramos, I.: The past, present and future of social networking and outsourcing: impact on theory and practice. In: UK Academy for Information Systems Conference Proceedings 2010: Information Systems: Past, Present and Looking to the Future (2010)
3. Brabham, D.: Crowdsourcing as a model for problem solving. Convergence: The International Journal of Research into New Media Technologies 14(1), 75 (2008)
4. Caprara, A., Monaci, M., Toth, P.: Models and algorithms for a staff scheduling problem. Math. Program. 98(1-3), 445–476 (2003)
5. Doan, A., Ramakrishnan, R., Halevy, A.Y.: Crowdsourcing systems on the world-wide web. Commun. ACM 54, 86–96 (2011)
6. Frankova, G., Séguran, M., Gilcher, F., Trabelsi, S., Dörflinger, J., Aiello, M.: Deriving business processes with service level agreements from early requirements. Journal of Systems and Software 84(8), 1351–1363 (2011)
7. Ipeirotis, P.: Demographics of mechanical turk. New York University, Tech. Rep. (2010)
8. Ipeirotis, P.G.: Analyzing the Amazon Mechanical Turk Marketplace. SSRN eLibrary 17(2), 16–21 (2010)
9. Karnin, E.D., Walach, E., Drory, T.: Crowdsourcing in the Document Processing Practice. In: Daniel, F., Facca, F.M. (eds.) ICWE 2010. LNCS, vol. 6385, pp. 408–411. Springer, Heidelberg (2010)
10. Kern, R., Thies, H., Satzger, G.: Statistical Quality Control for Human-Based Electronic Services. In: Maglio, P.P., Weske, M., Yang, J., Fantinato, M. (eds.) ICSOC 2010. LNCS, vol. 6470, pp. 243–257. Springer, Heidelberg (2010)
11. Kern, R., Zirpins, C., Agarwal, S.: Managing Quality of Human-Based eServices. In: Feuerlicht, G., Lamersdorf, W. (eds.) ICSOC 2008. LNCS, vol. 5472, pp. 304–309. Springer, Heidelberg (2009)
12. Khazankin, R., Psaier, H., Schall, D., Dustdar, S.: QoS-Based Task Scheduling in Crowdsourcing Environments. In: Kappel, G., Maamar, Z., Motahari-Nezhad, H.R. (eds.) ICSOC 2011. LNCS, vol. 7084, pp. 297–311. Springer, Heidelberg (2011)
13. Pinedo, M.: Scheduling: theory, algorithms, and systems. Springer (2008)
14. Psaier, H., Skopik, F., Schall, D., Dustdar, S.: Resource and agreement management in dynamic crowdcomputing environments. In: 2011 15th IEEE International Enterprise Distributed Object Computing Conference (EDOC), pp. 193–202 (September 2011)
15. Satzger, B., Psaier, H., Schall, D., Dustdar, S.: Stimulating Skill Evolution in Market-Based Crowdsourcing. In: Rinderle-Ma, S., Toumani, F., Wolf, K. (eds.) BPM 2011. LNCS, vol. 6896, pp. 66–82. Springer, Heidelberg (2011)