Available online at www.sciencedirect.com

ScienceDirect

Procedia Economics and Finance 32 (2015) 20 – 26

Procedia Economics and Finance

www.elsevier.com/locate/procedia

Emerging Markets Queries in Finance and Business

Realtime crowdsourcing with payment of idle workers in the Retainer Model

Andrea Ellero[a]*, Paola Ferretti[b], Giovanni Furlanetto[b]

[a]Department of Management, University Ca' Foscari of Venice, Fondamenta San Giobbe, Cannaregio 873, 30121 Venezia, Italy
[b]Department of Economics, University Ca' Foscari of Venice, Fondamenta San Giobbe, Cannaregio 873, 30121 Venezia, Italy

Abstract

The realtime applications of crowdsourcing are a very promising topic, due to its high potentialities, for example in marketing, security or telecommunication applications. Realtime crowdsourcing ensures that solutions to a given problem are obtained in the shortest possible time using collective intelligence. In order to be ready to carry out any requested task in realtime, crowdworkers must be available at any time. Here we focus on the payment of crowdworkers and on the trade-off between the expected waiting time for a task to be carried out and the number of workers in the pool that should not become too large otherwise the total cost increases. In particular we consider the, so called, Retainer Model in which crowdworkers are paid in order to be ready to carry out any requested task in realtime. The Retainer Model considers an expected total cost which takes into account both the amount paid to a crowdworker to be in idle-state and the loss when the task is not completed in realtime. After checking the existence of a minimum cost we characterize the optimal number of crowdworkers, and suggest a practical and quick way to obtain it. Moreover, we analyse the sensitivity of the optimal number of crowdworkers with respect to different task intensities.

© 2015 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).
Selection and peer-review under responsibility of Asociatia Grupul Roman de Cercetari in Finante Corporatiste

Keywords: crowdsourcing; Retainer Model; optimal number of workers

1. Introduction

The current meaning of the term Crowdsourcing is that of an online group activity in which knowledge, time and skills of the members of the group are gathered in order to undertake a common task. The principle is ancient: two brains are better than one and belongs to a sociological theory called the Wisdom of Crowds (Surowiecki, 2004). Crowdsourcing is a distributed model producing solutions for problems where users (the so-called crowd) are typically constituted around a community, based on web sites. They provide solutions or produce contents for the website. The crowd can also select the solution, finding the best among them all. The solution becomes property of the crowdsourcer: the requester who has posed the problem. Individuals in the crowd are usually remunerated, but

* Corresponding author. Tel.: +39-041-234-6930; fax: +39-041-234-8701.
E-mail address: ellero@unive.it

2212-5671 © 2015 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).
Selection and peer-review under responsibility of Asociatia Grupul Roman de Cercetari in Finante Corporatiste
doi:10.1016/S2212-5671(15)01359-3

sometimes they participate only for the intellectual stimulation or to be useful and help someone in producing a service.

Activities like collective designing of logos or web pages, crowdfunding, software developing and innovative solutions seeking are widespread on the Web and, although participation on crowdsourcing activities is often on a voluntary basis, it is also becoming more and more relevant for creating job opportunities, with all its pros and cons.

Crowdsourcing systems, indeed, make use of large amounts of Internet workers in a variety of jobs which range from extremely simple tasks, but which must be carried out very fast, like in Amazon's mechanical turk (mturk.com) or Minute Workers (minuteworkers.com), to the solution of rather tough problems or to realize large projects, like the Linus project (linus.com). Also the popular on-line encyclopedia Wikipedia (wikipedia.org) is the result of the contribution of countless web volunteers: by the way, Wikipedia contributors also provided a definition of the proper nature of crowdsourcing as "the process of obtaining needed services, ideas, or content by soliciting contributions from a large group of people, and especially from an online community, rather than from traditional employees or suppliers" (Crowdsourcing. In Wikipedia. Retrieved 9/9/2014, from en.wikipedia.org/wiki/Crowdsourcing).

One of the main current applications of crowdsourcing concerns the performance of microtasks, easy, simple and repetitive tasks to be carried out very fast like sorting or classifying photos and audio files, the transcription of scanned documents or the translation of short sentences. Another typical feature of real time applications of crowdsourcing is that the assigned tasks must start immediately upon request, and this is a very tough request.

The application of crowdsourcing in real time is something that the crowdsourcing world is still studying and trying to deal with, knowing its potentialities concerning the future of marketing, security and some phone applications that would simplify everyday life. Currently the speed reached to undertake a job in realtime is about 500 milliseconds, a very fast speed that has been reached applying the very recently defined Retainer Model (Bernstein et al., 2012), which is the model on which we focus in this paper. As a basis for comparison, consider that the human eye cannot detect more than about 25/30 frames per second in an animation film, i.e., with a frequency of one frame every 20 milliseconds discrete actions appear to be completely smooth.

The Retainer Model is the first and only one, that mathematically analyzes crowdsourcing in real time and considers the trade-off between the cost of hiring some workers to be prompt to immediately react when asked to perform a task and the Expected Waiting Time for a new incoming task. The existence of a minimum number of workers optimizing the Retainer Models trade-off and the study of the properties of the cost function is the aim of this paper.

The paper is organized as follows. In Section (2) we set the basic definitions for describing the Retainer Model which in Section (3) deals with two questions: how to optimize the response time minimizing the cost of the pool, given that the number of workers in the pool and the cost of having this pool are directly proportional, and how to maintain a short response time. Section (3) presents in particular the proof of the existence of a solution of the optimization problem and some of its properties, together with a recursive formula for its computation. Finally, in Section (4) we consider the problem of measuring how the candidate solutions change with reference to particular parameters, i.e., the mean number of arrivals, the retainer salary and the cost for a missed task. Section (5) ends the paper.

2. The Retainer Model

A crowdsourcing system runs when an incoming task finds some crowdworkers available to work on it. Incoming tasks create a list and this list is emptied on a first-come-first-served basis, task by task, by the crowdworkers. Bernstein et al. (2012), wondered whether it is possible to lower significantly the crowd latency, that is the period of time that affects a crowdsourcing system in the search for a new worker to answer a call in order to solve a task. The proposed model is the so-called Retainer Model. This model assumes that it is possible to engage crowdworkers in advance, leaving them to stay in a so called virtual pool. In this virtual pool they are paid also if no tasks are present and they are free to work on other things; but, when a task arises, they are obliged to put themselves to work. With this pool of speedy workers, the authors produced a series of experiments, using not only the concept of this Retainer Model, but also the rapid refinement algorithm. This rapid refinement algorithm is the process used by the authors to analyze the various responses to the experiments, gathered by the crowdworkers, and find the best answer trough them.

The Retainer Model addresses two questions: how to optimize the response time minimizing the cost of the pool (because the number of workers in the pool and the cost of having this pool are directly proportional) and how to maintain a short response time. The worker in the pool has to focus on the task as soon as it is asked. To do this, the best incentive is to pay an extra amount of money, allowing these workers to continue with other activities while waiting. The model must, therefore, ensure quick responses, be cost efficient (not wasting money unnecessarily hiring too many workers) and maintain the ability to respond after some time. A way that the authors found is to introduce an alert sound needed to awake the workers and put them to work immediately in front of the screen. The Authors even mention in the paper a new payment as a reward for the fastest reaction (but they do not describe it). The structure of the cost is very innovative and interesting, because the requester pays workers even when they are not working, compared with the pay to repeat old tasks at the top of the list of all tasks (which is the usual way to sort the tasks on websites such as Amazon Mechanical Turk (mturk.com/mturk/).

Bernstein et al. (2012) analyze the Retainer Model looking for the optimal number of workers to be employed in the virtual retainer pool. More precisely, they analyze the relationship between the *Expected Waiting Time* (the time that a requester has to wait when he commits a task to the crowd) and the *Total Cost* that the requester must sustain (maintaining a pool of workers) in the Retainer Model for realtime crowdsourcing. The Retainer Model is studied in the framework of Queueing Theory, in order to optimize the tradeoff between the Expected Waiting Time and the Total Cost of the retainer pool. The Retainer Model pays workers (put together in a common virtual pool) a small wage to respond on demand: the workers accept the task ahead and they are paid to keep the computer on. The mathematical model wants to predict how many workers should be employed to minimize this waiting time (while avoiding unnecessary costs). How does this model work? Upon the arrival of a task a worker leaves the pool, where it was plugged in, and works on the task. At this point, the system looks for another crowdworker to replace the one who is now busy with the task. The goal is to create a pool large enough to cope with the number of incoming tasks. So, increasing the number of crowdworkers in the pool minimizes the probability that the retainer pool becomes empty, avoiding the case where zero workers are available on new incoming task (and so avoiding the case of a possible non realtime answer). How to evaluate the arrival of the tasks and the time to process them?

For a queue system with n workers in parallel, the Load Coefficient ρ_l is defined as $\rho_l = \frac{E[t_s]}{E[t_a]n}$ where t_a represents the inter-arrival time between the customers and t_s represents the service time. By defining the average frequency of arrivals λ as $\lambda = \frac{1}{E[t_a]}$ and the maximum frequency of the service $\mu = \frac{1}{E[t_s]}$ it is $\rho_l = \frac{\lambda}{\mu n}$. For systems in which the customers can never be refused the stability condition is $0 \leq \rho_l < 1$. Crowdsourcing task arrival processes and retainer workers arrival processes are modeled as Poisson. In the Retainer Model the tasks arrive at the rate λ and the retainers workers answer at the rate μ after they are called. In this way, μ is thought by the authors as the time taken by a worker to process a task. The system then takes μ to re-fill a space left blank. In this model, the authors assume that λ and μ are known by the requester and that they do not change over time (or, if they change, this happens slowly). The traffic intensity ρ representing the percentage of system resources that are required to service newly incoming tasks is so defined: $\rho = \frac{\lambda}{\mu}$. Note again that $\rho < n$ is a necessary condition for the system, because otherwise the system is not able to refill the pool with new crowdworkers.

Thus, the trade off is between the probability that an incoming task should wait (the Expected Waiting Time) and the Total Cost of having a pool of n workers. In order to optimize this process, it is necessary to consider the probability that all the crowdworkers are working on the tasks and an incoming task should rest and wait (the so-called Probability of an Empty Pool): this is what should not happen in our realtime case. If on the one hand the probability of having an Empty Pool has to be minimized (by increasing the number of workers), on the other hand the number of workers in the pool cannot become too large because its cost.

3. Optimal number of workers

The probability of an Empty Pool is the probability that the virtual pool containing the crowdworkers waiting for the tasks is empty. Bernstein et al. (2012) supposed it follows the Erlang distribution, i.e. the probability of an Empty Pool $G(n)$ in a crowd of n workers is

$$G(n) = \frac{\frac{\rho^n}{n!}}{\sum_{i=0}^{n} \frac{\rho^i}{i!}}. \tag{1}$$

In this way, the *Expected Waiting Time* $EWT = \frac{G(n)}{\mu}$ sets a link between the number of crowdworkers n, the arrival rate of the tasks λ and the traffic intensity ρ. The cost of having a pool composed of n crowdworkers is represented by two main parts: the first is due to the number of employed workers, the second is related to the incapacity of the system to serve the requests in real time. In fact, all the workers in the pool who are waiting for an incoming task, even those who not engaged in a specific time, must receive a salary. The expected number of this portion of inactive crowdworkers who are still waiting for a task is $n - \rho[1 - G(n)]$; if s is the retainer salary, that is the salary to be paid because a crowdworker previously agree to keep the screen open and waiting for a task, thus staying in this pool, the first part of the cost per unit time on average is $s\{n - \rho[1 - G(n)]\}$. If C_{task} denotes the cost for a missed task, then the expected task cost is $C_{task}\, G(n)$: note that it is zero if the system is able to serve all the tasks in real time. Finally, the *Expected Total Cost* to the requester C_{tot} is so defined

$$C_{tot} = C_{task}\, G(n) + s\{n - \rho[1 - G(n)]\}. \tag{2}$$

Bernstein et al. (2012) main problem, was to find an optimal value of n in order to minimize C_{tot}, under the assumption that the requester knows λ and μ either through empirical observation or estimation. The Authors didn't face the problem of the existence of a solution of the optimization problem nor its characterization.

Here we focus our attention on these two aspects: after checking the existence of a minimum cost we characterize the optimal number of crowdworkers, and suggest a practical and quick way to obtain it. Let $C_{tot}(x)$ be the analytical extension in the nonnegative real variable x

$$C_{tot}(x) = C_{task}\, G(x) + s\{x - \rho[1 - G(x)]\} \tag{3}$$

where the strictly decreasing function

$$G(x) = \frac{1}{\int_0^{+\infty} e^{-\rho z}(1+z)^x\, dz} \tag{4}$$

satisfies the conditions $0 < G(x) \le 1$, $G(0) = 1$, $\lim_{x\to+\infty} G(x) = 0$. Given that (see for example Esteves et al., 1995)

$$G'(x) = -\frac{\int_0^{+\infty} e^{-\rho z}(1+z)^x \ln(1+z)\, dz}{\rho\left[\int_0^{+\infty} e^{-\rho z}(1+z)^x\, dz\right]^2} \tag{5}$$

it is

$$C'_{tot}(x) = 0 \iff \frac{\int_0^{+\infty} e^{-\rho z}(1+z)^x \ln(1+z)\, dz}{\rho\left[\int_0^{+\infty} e^{-\rho z}(1+z)^x\, dz\right]^2} = \frac{s}{C_{task} + s\rho}. \tag{6}$$

Moreover, the $C_{tot}(x)$ function results to be a strictly convex function of x if $x \ge 0$ given that

$$G''(x) = -2G(x)G'(x)\rho \int_0^{+\infty} e^{-\rho z}(1+z)^x \ln(1+z)\, dz - [G(x)]^2 \rho \int_0^{+\infty} e^{-\rho z}(1+z)^x [\ln(1+z)]^2\, dz \tag{7}$$

(see Esteves and Craveirinha, 2013) and

$$C''_{tot}(x) = G''(x)(C_{task} + s\rho). \tag{8}$$

With respect to extreme values on the domain, it is $C_{tot}(0) = C_{task}$ and $\lim_{x\to+\infty} C_{tot}(x) = +\infty$. In this way we can deduce that

1. the problem $\min_x C_{tot}(x)$ admits exactly one solution $x^* \in \mathbf{R}_+$;
2. the problem $\min_n C_{tot}(n)$ admits one or at most two solutions $n^* \in \mathbf{N}^*$.

Table 1. The total cost $C_{tot}(n)$ computed for different values of ρ ($s = 1$ and $C_{task} = 100$)

n	C_{tot} with $\rho = 0.5$	C_{tot} with $\rho = 1.0$	C_{tot} with $\rho = 2.5$
1	34.00	-	-
2	9.23	21.20	-
3	3.77	8.31	29.42
4	3.66	4.55	16.87
5	4.31	4.52	9.65
6	5.05	5.50	6.39
7	5.52	6.01	6.50
8	5.82	7.00	7.50
9	6.59	8.00	8.50
10	7.52	9.00	9.50
11	8.51	10.00	10.50
12	9.50	11.00	11.50

3.1. Computation of optimal solution

The proof that the problem of minimizing the Expected Total Cost function C_{tot} admits a solution and that there are at most two solutions opens a new question involving the computation of the optimal number of workers n^*. Clearly, it depends on the choice of some exogenous parameters: ρ, s and C_{task}. Given strict convexity of the function C_{tot} and the limit results $C_{tot}(0) = C_{task} > 0$ and $\lim_{x\to+\infty} C_{tot}(x) = +\infty$, it is possible to detect the minimum point or the two equivalent minimum points by iteratively computing the objective function that has to be minimized. The computation stops when $C_{tot}(n) \leq C_{tot}(n+1)$: if $C_{tot}(n) < C_{tot}(n+1)$ then $n^* = n$, otherwise n and $n+1$ are the two equivalent solutions. On the following formula (see Jagerman, 1984)

$$[G(n+1)]^{-1} = \frac{n+1}{\rho}[G(n)]^{-1} + 1 \tag{9}$$

is essentially based the recursive computation of the Expected Total Cost $C_{tot} = C_{tot}(n)$:

$$C_{tot}(n+1) = C_{tot}(n) + \frac{G(n)[\rho - (n+1) - \rho G(n)]}{n+1+\rho G(n)}(C_{task} + s\rho) + s \tag{10}$$

on which the minimum valued (and the optimal solution/s) can be computed.

4. Sensitivity analysis

The Expected Total Cost function C_{tot} defined in (2) depends on the choice of the number n of workers, the traffic intensity ρ, the retainer salary s and the missed task cost C_{task}. The probability of an empty pool $G(n)$ varies depending on the parameter ρ and it strictly increases (see Esteves and Craveirinha, 2013). This implies that the $n^* = \arg\max C_{tot}(n)$ may depend on the choice of ρ (see table 1).

Starting from equation (6) it is possible to study how an optimal number of workers in the crowd x^* changes with reference to the exogenous variables ρ, s, C_{task}. In other words, the First Order Condition (6), that in this problem is sufficient to characterise the solution, allows the analysis of an optimal crowd as function of exogenous variables like the mean number of arrivals, the retainer salary and the cost for a missed task. Let us consider the equation

$$F(x,\rho,s,C_{task}) = -\frac{\int_0^{+\infty} e^{-\rho z}(1+z)^x \ln(1+z)\,dz}{\rho\left[\int_0^{+\infty} e^{-\rho z}(1+z)^x\,dz\right]^2}(C_{task} + s\rho) + s = 0. \tag{11}$$

By the implicit function theorem it is possible to study the dependence of the optimal solution x^* as a function of the exogenous variables ρ, s, C_{task}. More precisely, let us consider for any $(x,\rho) \in \mathbf{R} \times \mathbf{R}_+$ the infinitely differentiable function $G(x,\rho)$

$$G(x,\rho) = \frac{1}{\int_0^{+\infty} e^{-\rho z}(1+z)^x dz} \tag{12}$$

with $G'_x(x,\rho)$ given in (5) and

$$G'_\rho(x,\rho) = \left(\frac{x}{\rho} - 1 + G(x,\rho)\right) G(x,\rho). \tag{13}$$

Then

$$\frac{\partial x}{\partial \rho} = -\frac{G''_{x\rho}(x,\rho)(C_{task}+s\rho) + sG'_x(x,\rho)}{G''_{xx}(x,\rho)(C_{task}+s\rho)} \tag{14}$$

where $G''_{x\rho}(x,\rho) = (G'_\rho)'_x(x,\rho) < 0$, given that $G'_\rho(x,\rho)$ is a strictly decreasing function of x when $G'_x(x,\rho) < -1/\rho$: in fact, $G'_\rho(x,\rho)$ is product of two strictly decreasing nonnegative functions of x. With reference to (14) it is possible to deduce that the optimal number x^* is a strictly increasing function of the traffic intensity ρ. Again it is

$$\frac{\partial x}{\partial s} = -\frac{1+\rho G'_x(x,\rho)}{G''_{xx}(x,\rho)(C_{task}+s\rho)}, \tag{15}$$

$$\frac{\partial x}{\partial C_{task}} = -\frac{G'_x(x,\rho)}{G''_{xx}(x,\rho)(C_{task}+s\rho)}. \tag{16}$$

Looking at the previous partial derivatives, one can make the following remarks concerning the (strict) monotonic behavior of the optimal numbers of workers in the crowd:

1. when the traffic intensity increases the optimal number of workers in the pool increases too;
2. if the retainer salary becomes higher then the number of workers in the pool minimizing the total expected cost increases;
3. when the cost for a missed task rises, necessarily the optimal number of crowd workers increases

given that $G'_x(x,\rho) < -1/\rho$.

Note that the percentage change of $C_{tot}(x)$ divided by the percentage change of the input x, i.e. the partial output elasticity, is

$$E_{C_{tot}}(x) = x\frac{G'(x)(C_{task}+s\rho)+s}{C_{task}\,G(x)+s\{x-\rho[1-G(x)]\}}. \tag{17}$$

5. Concluding remarks

Crowdsourcing is based on collaboration among persons belonging to the same network: hiring crowdworkers allows to solve tough tasks. Realtime crowdsourcing, in particular, gives answers or executes tasks within few seconds. We study the analytical properties of the Total Cost function, as proposed in the Retainer Model, and prove the existence of its minimum. The minimum point can be computed by means of a recursive approach, obtaining the optimal number of workers to be hired. We also consider the sensitivity of the optimal number of crowdworkers with respect to the parameters of the model, like traffic intensity and workers salary.
Future work should be devoted to the analysis of realtime crowdsourcing with real data taken from a website application, putting special attention to the real cost of missed tasks, at least as a (convincing) range of values.

References

[1] Surowiecki K. (2004) The wisdom of crowds. New York: Doubleday.

[2] Bernstein MS, Karger DR, Miller RC, Brandt J. Analytic Methods for Optimizing Realtime Crowdsourcing. CoRR 2012;abs/1204.2995. http://arxiv.org/abs/1204.2995.

[3] Esteves JS, Craveirinha J, Cardoso D. Computing erlang-*b* function derivatives in the number of servers. Communications in Statistics. Stochastic Models 1995;11,2:311-331.

[4] Esteves JS, Craveirinha J. On a bicriterion server allocation problem in a multidimensional Erlang loss system. Journal of Computational and Applied Mathematics 2013;252:103-119.

[5] Jagerman DL. Methods in traffic calculations. AT&T Bell Laboratories Technical Journal 1984;63,7:12831310.