

A Probabilistic Approach to Crowdsourcing Pareto-Optimal Object Finding by Pairwise Comparisons

NIGESH SHAKYA, The University of Texas at Arlington
CHENGKAI LI, The University of Texas at Arlington

This is an extended study on crowdsourcing Pareto-Optimal Object Finding by Pairwise Comparisons. The prior study on the same topic demonstrate the framework and algorithms used to determine all the Pareto-Optimal objects with the goal of asking the fewest possible questions to the crowd. One of the drawbacks in that approach is it fails to incorporate every inputs given by the crowd and is biased towards the majority. We have developed an approach which represent the inputs provided by users as probabilistic values rather than a concrete one.

The goal of this study is to find the ranks of the objects based on their probability of being Pareto-Optimal by asking every possible questions. We have used the possible world notion to compute these ranks. Further we have also demonstrated the prospect of using Slack (a cloud-based team collaboration tool) as a Crowdsourcing platform.

1 INTRODUCTION

The size of the crowd is so much important to extract the wisdom of the crowd. The aggregation of diverse opinions of the participants helps in reducing errors and noise in the collective knowledge accumulated. However, the presence of dissenting opinions may have considerable hindrances, thus resulting in very unstable collective performance across problems [3]. These problems come to surface due to the collective size, the transferability of expertise and the task difficulty. In the prior study [1] this issue was tackled by maintaining a certain threshold of tolerability so that a consensus is reached only when the response from the crowd has reached above that threshold and consequently ignoring the differing opinions. The purpose of this study is to take into account the minority opinions to check if incorporating the minority opinions improves the wisdom of the crowd.

In the prior study [1] has already provided the concept of Pareto-Optimal Objects. It is very essential to understand the concept of Pareto-Optimal objects to understand the core value of our work. Consider a set of objects O , a set of criteria C for comparing the objects and a crowd participation size S . An object $x \in O$ is Pareto-Optimal if and only if x is not dominated by any other object. Object y dominates x (denoted $y > x$) if and only if x is not better than y by any criteria and y is better than x by at least one criterion, i.e. $\forall c \in C : x \not>_c y$ and $\exists c \in C : y >_c x$. If x and y do not dominate each other (i.e., $x \not> y$ and $y \not> x$), we denote it by $x \sim y$. The difference between [1] and this paper exists in the representation of the *preference relation* between the pair over a criterion as a probabilistic values of "is better than with regard to criterion $c(>)$ ", "is indifferent regarding $c(\sim)$ " and "is not better than regarding $c(<)$ ".

Consider n objects, r criteria and Q pairwise comparisons on all object pairs by every criterion which lead to $r \cdot n \cdot (n-1)/2$ comparisons. Every such comparison can be represented as a probability value for $>, \sim, <$ relations, i.e.

$$r_{lt}(x?y) = \left\{ (x?y) \in Q : \exists Pr(x?Ry) \forall R \in \{>, \sim, <\}, \text{ where } Pr(x?Ry) = \frac{\# \text{ users who voted for } (x?Ry)}{S.length} \right\} \quad (1)$$

Several similar studies on the related topic have been previously conducted. Dorte Lerche et al. [4] find ranks of probability for partial orders by projecting partial orders into total order. Our work differ from them in that we have used the partial orders in its default order without converting into linear extensions. Further, Adrianos Papamarcou et al. [2] points out the importance of lower probability in representing uncertainty. However, it does not relate to Crowdsourcing and Partial Ordering.

The new methods approached by this paper are summarized and explained as below.

- The sample size of the crowd for each pairwise comparison question is finalized based on the desired confidence interval instead of selecting some arbitrary crowd size.
- We utilize the theory of possible worlds to exhaustively capture all the possible cases of responses for each question instead of assimilating a concrete response generated as a result of using a threshold value to generalize the consensus of the crowd.
- Instead of using a popular crowdsourcing Internet marketplace tool like Amazon Mechanical Turk (MTurk) we devise a "Slack" based bot to capture the crowd inputs with the intent of sampling data for homogeneous decision making. Another incentive to have a closed group decision making is the flexibility to choose the crowd based on their expertise in the related task and there is a higher probability of getting the accurate inputs from the crowd.

2 FRAMEWORK

The representation of outcomes in probability values as shown in Equation 1 inhibits skipping any questions as we shall show later. The entire problem set can be broken down into subproblems of finding confident enough samples size and the problem of finding possible worlds with pruning included.

2.1 Find the Confident Sample Size

One of the hurdles in crowdsourcing is to get the wisdom of the entire population using a sample size. It is practically impossible to get the actual proportion of the crowd favoring in our desired characteristic due to the enormous cost associated with collecting data from the entire population. We need to be confident enough to postulate that the sample size we have used for data collection is the representative of the whole population. Instead of finding the exact true proportion we estimate the true proportion by calculating a range of proportions for each characteristic with certain level of confidence($\alpha\%$) as shown in Fig. 1.

The first step is to calculate the minimal sample size(MSS). Minimal sample size for a certain $\alpha\%$ confidence level is the least crowd size that needs to be sampled.

$$n = \frac{N \cdot \hat{p} \cdot (1 - \hat{p})}{(N - 1) \cdot \frac{d^2}{Z_{\alpha/2}^2} + \hat{p} \cdot (1 - \hat{p})} \quad (2)$$

where N is the total finite population size, \hat{p} is the estimated sample proportion distribution for desired characteristic, d is the margin of error and $Z_{\alpha/2}$ is the Z -index for α -confidence level [6]. We assume that there is an equal distribution of proportions across all three replies ($>$, \sim , $<$) since we don't know the true proportion. As a result the value of \hat{p} will be $1/3$. This assumption is a conservative approach and is helpful when no such past data collection phase with the similar task was conducted. Another way to estimating \hat{p} is using an educated guess where we don't assume an equal distribution and use the \hat{p} value from the result of such past experiments.

The next step is the calculation of confidence intervals for the population proportion.

$$\hat{p} \pm \text{Margin of Error(M.E.)} \quad (3)$$

where \hat{p} is the proportion of the sample population in favor of the desired characteristic and *Margin of Error(M.E.)* is the difference of interval endpoint to the true proportion. M.E. can be formulated as

$$\text{Margin of Error(M.E.)} = Z_{\alpha/2} \cdot \left(\sqrt{\frac{\hat{p} \cdot (1 - \hat{p})}{n}} \right) \quad (4)$$

where n is the sample size. Consider a set of objects $x, y \in O$ and a set of criteria $c \in C$ for comparing objects. The confidence intervals must be calculated for all three types of proportions of our interest:

- (1) Confidence Interval ($CI_{>}$) for the proportion ($\hat{p}_{>}$) of the crowd who favors $x >_c y$
- (2) Confidence Interval (CI_{\sim}) for the proportion (\hat{p}_{\sim}) of the crowd who favors $x \sim_c y$
- (3) Confidence Interval ($CI_{<}$) for the proportion ($\hat{p}_{<}$) of the crowd who favors $x <_c y$

Consider a certain threshold $t \in [0, 1]$. We then check if all three CIs ($CI_{>}$, CI_{\sim} , $CI_{<}$) are less than t . If any of these three confidence intervals are more than t , it signifies that we are not confident enough with the proportions. As a result, we increase the crowd sample size n and find the confidence intervals again. We repeat until all three CIs have values less than or equal to t as shown in Algorithm 1 .

ALGORITHM 1: Find the proportions with α -confidence level

Input: A minimal sample size m , a certain tolerable threshold t and total population size N

Output: Accepted proportions of the crowd $p_{>}$, p_{\sim} , $p_{<}$

$n \leftarrow m - 1$

while $n = m - 1$ **or** $range(CI_{>}) > t$ **or** $range(CI_{\sim}) > t$ **or** $range(CI_{<}) > t$ **do**

$n \leftarrow n + 1$

$p_{>}, p_{\sim}, p_{<} \leftarrow \text{findProportions}(n)$

$CI_{>} \leftarrow \text{findCI}(p_{>}, n, \alpha)$

$CI_{\sim} \leftarrow \text{findCI}(p_{\sim}, n, \alpha)$

$CI_{<} \leftarrow \text{findCI}(p_{<}, n, \alpha)$

if $n = N$ **then**

| break

end

end

return $p_{>}, p_{\sim}, p_{<}$

Further, it is also possible that the range of the confidence interval might be zero because of zero valued proportion. The zero proportions occur when none of the crowd respond in favor of that characteristic. We use a simple "Rule of Three" [5] to resolve this issue. We simply replace the upper limit of the confidence interval with $3/n$. It is notable that as the crowd sample size n increase the value decreases.

Ideally, upon increasing the sampling crowd size the range of confidence interval reduces. A tolerable low range of CI signifies we are $\alpha\%$ confident that the obtained proportion captures the true proportion of the desired characteristic within that interval. In other words, if we perform the same data collection over and over again one hundred times, it is α times likely that we shall get the range of CI which captures the true proportion.

2.2 Possible Worlds Theory

By using probability to incorporate every opinion of the crowd, we need to consider every possible scenario of outcomes. A "world" is symbolic representation of any of such scenario of outcomes. Diverting from the previous study, our partial order graphs are complete as we have a complete knowledge of the crowd preference relations for various criteria. Given n objects and r criteria, the total number of pairwise comparisons on all objects by every criterion would be $r.n.(n-1)/2$. Note that this is the brute-force approach. It is not possible to optimize the number of pairwise comparison questions because we cannot skip any such comparisons. In the prior study [1] the property of transitive closure of outcomes enable skipping comparisons. However, transitivity is not valid in our case. Consider $x, y, z \in O$ and $c \in C$. If $\text{rlt}(x ?_c y) = \{p_{x>_c y}, p_{x\sim_c y}, p_{x<_c y}\}$ and $\text{rlt}(y ?_c z) = \{p_{y>_c z}, p_{y\sim_c z}, p_{y<_c z}\}$. Based on the outcomes above we see $p_{x>_c y}$ and $p_{y>_c z}$. We might be tempted in assuming that $p_{x>_c z}$ holds by applying transitive closure and hence wanting to skip the calculation for the comparison between x and z . However, if we look closely, the existence of $p_{x<_c z}$ is also likely given the probabilities $p_{x<_c y}$ and $p_{y<_c z}$. Similarly, $p_{x\sim_c z}$ is also probable. Hence, we can't skip the comparison between x and z .

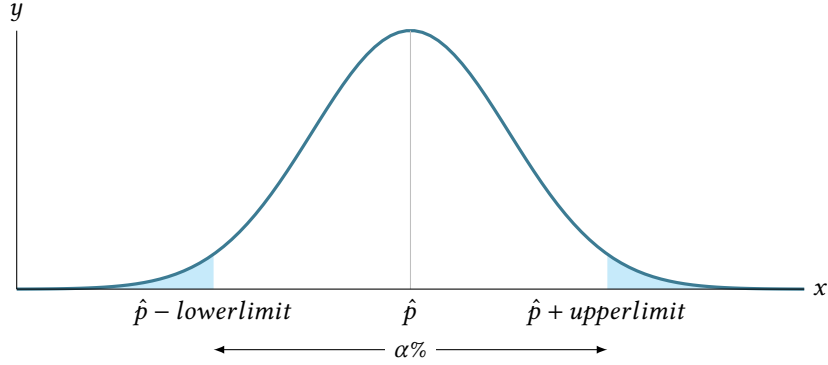


Fig. 1. Bell Curve for estimating population proportion with $\alpha\%$ confidence level

The possible world theory can be subdivided into four steps which will lead to the ranks of the objects:

(1) **List all the possible worlds**

The number of possible worlds is the Cartesian product of the results of all the pairwise comparison questions. Consider i unique comparison questions Q_1, \dots, Q_i and each question Q_j have three outcomes of ($>, \sim, <$). The possible worlds can be calculated as

$$Q_1 \times \dots \times Q_i = \{(u_1, \dots, u_i) : u_j \in Q_j\} \quad (5)$$

where (u_1, \dots, u_i) is a world and each u_j can have the probability of either $p_{>}, p_{\sim}$ or $p_{<}$

(2) **Calculate the probability of existence of every world**

Every possible world might have varying probability of existence (p_{exist}). The p_{exist} of a world can be calculated as the product of all the probability outcomes inside that world. Consider a world $u = (u_1, \dots, u_i)$ with the respective probability of its comparison elements as (p_1, \dots, p_i) , the p_{exist} can be computed as

$$p_{exist} = \prod_{j=1}^i p_j \quad (6)$$

Intuitively, a world with low p_{exist} means that the combination of outcomes inside that world is least probable and vice versa. It is intuitive enough to state that as the comparison questions increases the value p_{exist} gets smaller and vice versa.

(3) **Find the Pareto-Optimal objects in each world**

We need to exhaustively go through every possible world to compute their respective Pareto-Optimal objects. Once we get the Pareto-Optimal objects we assign the ranks for those objects in that particular world. The ranks are assigned the same value as the p_{exist} of that world. Every Pareto-Optimal objects have same rank because the probability of every Pareto-Optimal objects is equal for that particular world. For all the non Pareto-Optimal objects we assign the rank of those objects as zero value. There can also be the cases where no Pareto-Optimal objects are found because every object dominates every other objects. In such a case we introduce a variable χ used to represent worlds that have contradiction. The rank of χ also equals to p_{exist} if the contradiction exists otherwise it will be zero. Consider a set of Pareto-Optimal objects $v \in O$ and a set of non Pareto-Optimal objects $w \in O$ in a world u .

$$rank_{y,u} = \left\{ \begin{array}{ll} p_{exist}, & \text{for } y \in v \\ 0, & \text{for } y \in w \end{array} \right\}, \quad rank_{\chi,u} = \left\{ \begin{array}{ll} p_{exist}, & \text{for } v = \emptyset \\ 0, & \text{for } v \neq \emptyset \end{array} \right\}$$

$$\begin{array}{lll}
p_{exist}(PW_{16}) = p_2 * p_6 * p_7 & p_{exist}(PW_{17}) = p_2 * p_6 * p_8 & p_{exist}(PW_{18}) = p_2 * p_6 * p_9 \\
p_{exist}(PW_{19}) = p_3 * p_4 * p_7 & p_{exist}(PW_{20}) = p_3 * p_4 * p_8 & p_{exist}(PW_{21}) = p_3 * p_4 * p_9 \\
p_{exist}(PW_{22}) = p_3 * p_5 * p_7 & p_{exist}(PW_{23}) = p_3 * p_5 * p_8 & p_{exist}(PW_{24}) = p_3 * p_5 * p_9 \\
p_{exist}(PW_{25}) = p_3 * p_6 * p_7 & p_{exist}(PW_{26}) = p_3 * p_6 * p_8 & p_{exist}(PW_{27}) = p_3 * p_6 * p_9
\end{array}$$

Step 3: Find the Pareto-Optimal Objects in each Possible World and assign the respective ranks in the same order of objects as

$$\begin{bmatrix} x \\ y \\ z \\ \chi \end{bmatrix}$$

There will be 27 different ranks

$$\begin{array}{llllll}
rank_1 = \begin{bmatrix} p_1 * p_4 * p_7 \\ 0 \\ 0 \\ 0 \end{bmatrix} & rank_2 = \begin{bmatrix} p_1 * p_4 * p_8 \\ 0 \\ 0 \\ 0 \end{bmatrix} & rank_3 = \begin{bmatrix} p_1 * p_4 * p_9 \\ 0 \\ 0 \\ 0 \end{bmatrix} & rank_4 = \begin{bmatrix} p_1 * p_5 * p_7 \\ 0 \\ 0 \\ 0 \end{bmatrix} & rank_5 = \begin{bmatrix} p_1 * p_5 * p_8 \\ 0 \\ p_1 * p_5 * p_8 \\ 0 \end{bmatrix} \\
rank_6 = \begin{bmatrix} p_1 * p_5 * p_9 \\ 0 \\ p_1 * p_5 * p_9 \\ 0 \end{bmatrix} & rank_7 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ p_1 * p_6 * p_7 \end{bmatrix} & rank_8 = \begin{bmatrix} 0 \\ 0 \\ p_1 * p_6 * p_8 \\ 0 \end{bmatrix} & rank_9 = \begin{bmatrix} 0 \\ 0 \\ p_1 * p_6 * p_9 \\ 0 \end{bmatrix} & rank_{10} = \begin{bmatrix} p_2 * p_4 * p_7 \\ p_1 * p_5 * p_9 \\ 0 \\ 0 \end{bmatrix} \\
rank_{11} = \begin{bmatrix} p_2 * p_4 * p_8 \\ p_2 * p_4 * p_8 \\ 0 \\ 0 \end{bmatrix} & rank_{12} = \begin{bmatrix} p_2 * p_4 * p_9 \\ 0 \\ 0 \\ 0 \end{bmatrix} & rank_{13} = \begin{bmatrix} p_2 * p_5 * p_7 \\ p_2 * p_5 * p_7 \\ 0 \\ 0 \end{bmatrix} & rank_{14} = \begin{bmatrix} p_2 * p_5 * p_8 \\ p_2 * p_5 * p_8 \\ p_2 * p_5 * p_8 \\ 0 \end{bmatrix} & rank_{15} = \begin{bmatrix} p_2 * p_5 * p_9 \\ 0 \\ p_2 * p_5 * p_9 \\ 0 \end{bmatrix} \\
rank_{16} = \begin{bmatrix} 0 \\ p_2 * p_6 * p_7 \\ 0 \\ 0 \end{bmatrix} & rank_{17} = \begin{bmatrix} 0 \\ p_2 * p_6 * p_8 \\ p_2 * p_6 * p_8 \\ 0 \end{bmatrix} & rank_{18} = \begin{bmatrix} 0 \\ 0 \\ p_2 * p_6 * p_9 \\ 0 \end{bmatrix} & rank_{19} = \begin{bmatrix} 0 \\ p_3 * p_4 * p_7 \\ 0 \\ 0 \end{bmatrix} & rank_{20} = \begin{bmatrix} 0 \\ p_3 * p_4 * p_8 \\ 0 \\ 0 \end{bmatrix} \\
rank_{21} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ p_3 * p_4 * p_9 \end{bmatrix} & rank_{22} = \begin{bmatrix} 0 \\ p_3 * p_5 * p_7 \\ 0 \\ 0 \end{bmatrix} & rank_{23} = \begin{bmatrix} 0 \\ p_3 * p_5 * p_8 \\ p_3 * p_5 * p_8 \\ 0 \end{bmatrix} & rank_{24} = \begin{bmatrix} 0 \\ 0 \\ p_3 * p_5 * p_9 \\ 0 \end{bmatrix} & rank_{25} = \begin{bmatrix} 0 \\ p_3 * p_6 * p_7 \\ 0 \\ 0 \end{bmatrix} \\
rank_{26} = \begin{bmatrix} 0 \\ p_3 * p_6 * p_8 \\ p_3 * p_6 * p_8 \\ 0 \end{bmatrix} & rank_{27} = \begin{bmatrix} 0 \\ 0 \\ p_3 * p_6 * p_9 \\ 0 \end{bmatrix}
\end{array}$$

Step 4 : Calculate the final rank by adding up all the rank matrices

$$rank = ranks_1 + ranks_2 + \dots + ranks_{27} = \begin{bmatrix} p_1 * p_4 * p_7 + p_1 * p_4 * p_8 + \dots + 0 + 0 \\ 0 + \dots + p_1 * p_5 * p_9 + \dots + p_3 * p_6 * p_8 + 0 \\ 0 + 0 + \dots + p_1 * p_5 * p_8 + \dots + p_3 * p_6 * p_9 \\ 0 + \dots + p_1 * p_6 * p_7 + \dots + p_3 * p_4 * p_9 + \dots + 0 \end{bmatrix} = \begin{bmatrix} \rho_x \\ \rho_y \\ \rho_z \\ \rho_\chi \end{bmatrix}$$

where $\rho \in [0, 1]$ is the final rank

2.4 Setbacks

With the Possible World Theory the problem space increases exponentially for every pairwise comparison. Essentially, the problem space denotes the resources required to capture all the total number of possible worlds. The Brute-Force approach that we have adopted is very expensive in terms of computation. Given a set of n objects and r criteria, the complexity of the problem grows at the rate of $O(3^{r \cdot n \cdot (n-1)/2})$. The base value three is the outcome size ($>, \sim, <$) and the root is the total number of pairwise comparisons. Since the total number of pairwise comparisons also grows at the polynomial rate the problem growth rate is polynomially exponential. For instance, for 5 objects and 2 criteria the total number of possible worlds is 3486784401 which is already a very heavy number to compute. In real world the number of objects to compare and the criteria to judge them could be a very large finite number. Pragmatically, there is no way to compute such a large volume of processing without exhausting all the resources and time. With the growth rate that big the problem of memory and execution time also increases at the same rate.

2.5 Zero World Pruning

There is a way to mitigate the computational requirements with the growth rate. The concept of Zero World Pruning suggests that if the p_{exist} is very low or equal to zero, it will contribute significantly less or none towards the final rank of the objects on being Pareto-Optimal. Hence, we can skip the computation to calculate Pareto-Optimal objects in those zero worlds. The scope of the problem redirects towards finding the possible worlds with zero p_{exist} so as to skip calculation of the Pareto-Optimal objects in those world.

Consider a threshold value $s \approx 0$. There are 2 different approaches to Zero World Pruning:

(1) Proportion Pruning

If any of the proportion p of an outcome of a pairwise comparison is very close or equal to zero we can skip the computational requirement for all of the worlds which have that proportion. The pruning rate of the possible worlds decrease as the number of the such zero proportion increases. For every pairwise comparison there can be at most two zero proportions. Given the number of zero proportions the total number of pruned worlds can be found using the Algorithm 2

ALGORITHM 2: Find the Total number of Pruned worlds using Proportion Pruning

Input: A set of zero proportions ZP_r for every i th result of pairwise comparisons $r \in R_1, \dots, R_i$

Output: Total number of pruned worlds

$n \leftarrow R.length$

$prunedWorld \leftarrow 0, rem \leftarrow 3^n$

for $r \leftarrow R_1$ **to** R_i **do**

$pList \leftarrow r.getZeroProportions()$

$b \leftarrow 0$

foreach $p \leftarrow pList$ **do**

$b \leftarrow b + rem/3$

end

$prunedWorld \leftarrow prunedWorld + b$

$rem \leftarrow 3^n - prunedWorld$

end

return $prunedWorld$

(2) Marginal Pruning

If the p_{exist} of any of the possible worlds is approximately close or equal to zero we can skip the further

computations for all those worlds. The threshold for this pruning should be much closer to zero than the threshold used for Proportion Pruning since the p_{exist} of a world is much smaller than its constituent proportions. We can derive the new threshold value \hat{s} using the original threshold s . Consider the total number of possible worlds n .

$$\hat{s} = \log(s * 1/n) \tag{8}$$

We then check to see if the $\log(p_{exist})$ of the world $< \hat{s}$ and prune the unnecessary computation for those worlds. Since the new threshold value \hat{s} can get extremely small, the use of logscale makes the computation much convenient.

One interesting attribute of this pruning approach is it breaks the relationship between the number of zero proportions and the total number of zero worlds. One positive way to look at it is when the zero proportions are just slightly above the threshold s and the number of possible worlds increase, the p_{exist} can still be very less and thus those worlds can be pruned.

3 EXPERIMENTS

We designed and conducted experiments using both a real crowdsourcing platform and simulations based on a real dataset.

(1) Slack as a Crowdsourcing tool

Slack is a cloud based team collaboration tool which can be used mostly for exchanging instant messages between parties through different forms like direct messages, channels or groups. It has gained a huge popularity in recent days and many organizations are adopting this technology because it is highly flexible in terms of the availability of a lot of custom integrations as well as the convenience in creating our own customization suited to the organizational needs. One can create bot users for some specific purposes and add them to the team to automate some mundane tasks.

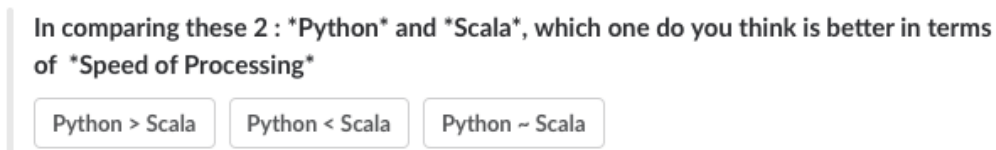












Fig. 2. Question format in Slack



Fig. 3. Question Respond format

In this study we have studied the prospect of using Slack as a crowdsourcing tool in decision making. The motivation for conducting experiments using Slack is to get to a decision for certain tasks within a team provided certain comparison objects and criteria to compare them. For instance, a CEO of a company might be interested in knowing the best place to relocate the address of the company based on the preferences of the employees. Instead of using a traditional voting mechanism, a Slack integration in the form of a bot might handle that task with ease by using less time and resources.

The Slack bot that we have developed asks questions (Figure 2) to the team members and based on their responses (Figure 3) run the Possible World algorithm to get the ranks of the objects O . There are

Name	Description		
/ask	Ask qstns -> channel users		
/settimeout	Sets the timeout for question		
/helpme	Launches the help section		
/stopall	Stops all process		
/getresults	Find the crowdsourcing result		

Create New Command

Copy Existing Command

Fig. 4. Commands that are currently recognized by the bot

specific commands that can be detected by our bot (Figure 4). The detailed procedure through which our bot operates is shown in as Algorithm 3. For each of the pairwise comparison question q we ask it to the unique team members m until the confident enough sample size has been collected. The sample size for each q might be different. There is a MAX WAIT TIME which is the waiting time for the team members to respond to the questions. If we don't receive reply for questions from the team members within that time interval we again continue the procedure until all the questions are exhausted. A question can be exhausted when the sample size collected is confident enough. The maximum number of questions that can be answered by a team member is the size of the questions. This is to say that no team members will answer to the same question twice. Once we have confident enough sample size for all of the questions, we run the Possible World Algorithm to get the ranks of the objects O .

(2) **Data Collection Phase**

We conducted the data collection within our IDIR lab setting amongst the 15 scholars who simultaneously participated by answering to the questions. The task was to find the rank of different Programming Languages provided the objects size of 8 and 4 different criteria. For the purpose of experimentation we used the Brute Force approach to collect inputs. Instead of using the Algorithm 3 to collect data we just use exhaustive approach to ask all the participants all the pairwise comparison objects. By doing so every participants will respond to exactly the total number of pairwise comparison questions.

(3) **Experiment by Simulation**

The real data collected is used in various ways to simulate different circumstances to conduct performance and feasibility evaluation of the algorithm. Throughout our experiments we have assumed 95% Confidence Level. Our first experiment was to check under what parameters the minimal sample size(MSS) changes. As shown in Figure 5 the MSS tends to grow steadily as the total population size grows for low Margin of Error(MOE). However, by keeping a broad threshold for MOE the MSS tends to stay at lower value for increasing population size.

The next experiment was to check the final sample size(FSS) when we are confident enough about the outcomes. By keeping the fixed total population size of 15, we change the MOE on various number of

ALGORITHM 3: Data Collection Procedure using Slack

Input: A set of pairwise comparison questions Q_1, \dots, Q_i , team members M_1, \dots, M_i

Output: Ranks of the objects O

```

qList ← Q1 to Qi
while qList.length > 0 and wait_time > MAX_WAIT_TIME do
  mList ← M1 to Mi
  foreach q ← qList do
    b ← q.sampleSize
    while mList.length > 0 do
      m ← mList.pop()
      (q, m) ← question-member pairing
      SLACK.ask(q, m)
      if b >= MINIMAL SAMPLE SIZE then
        break
      end
    end
  end
  end
  /* wait for members to respond */
  qList ← SLACK.getResponses()
  foreach q ← qList do
    if q.isConfidentSampleSize then
      qList.pop()
    end
  end
end
end
ranks ← runParallelWorldAlgorithm(qList)
return ranks
  
```

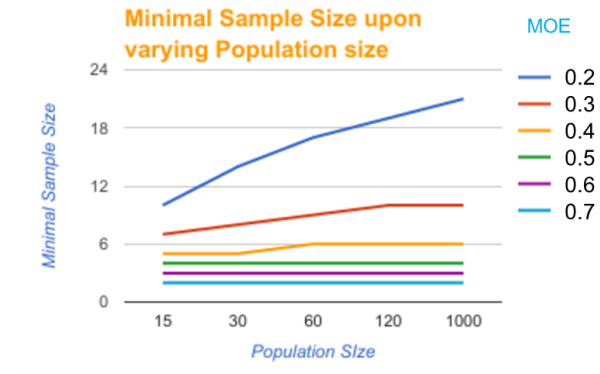


Fig. 5. Minimal Sample Size upon varying Total Population and Margin of Error

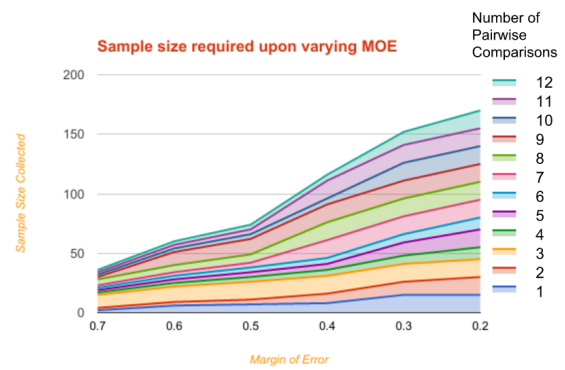


Fig. 6. Final Sample Size required upon varying MOE and number of pairwise comparisons

pairwise comparisons(n). It is obvious that for low MOE the FSS grows on each set of pairwise comparison. For one of the pairwise comparison($n=3$) we can see that the FSS is significantly larger than for the rest. The explanation for this is that we are not confident enough in the FSS even when we have sampled through all 15 users. If such a case arise, having limited number of total population we simply stop and accept the outcomes no matter whatever confidence intervals we get. This explanation is also sufficient to understand why most of the FSS for low MOE tend to be around the same value. As the MOE decreases FSS of 15 is also not sufficient for them and we need more sample size.

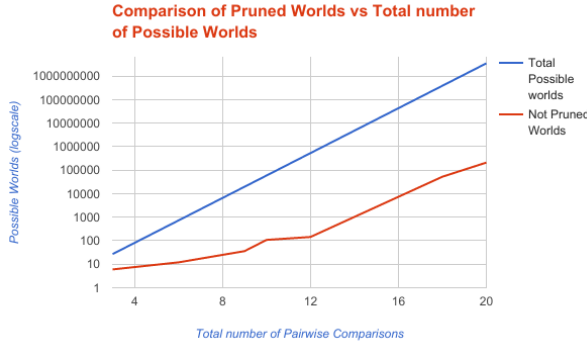


Fig. 7. Possible Worlds that needs to computed for Pareto-Optimal Objects upon varying number of pairwise comparisons

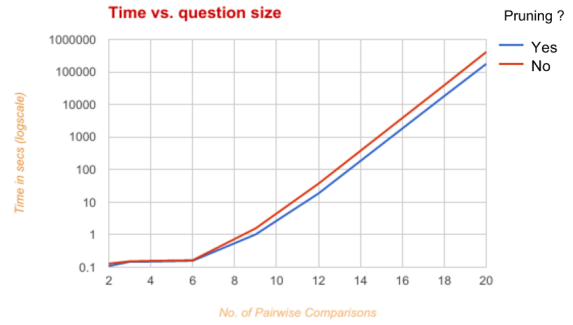


Fig. 8. Computational time taken for varying number of pairwise comparison

Figure 7 is another experiment to demonstrates the relationship between number of pairwise comparisons and the growth rate of Possible worlds which actually need to be calculated to find the Pareto-Optimal objects. As seen in the Figure the growth rate seems to be decreasing for such worlds. The reason for its decrease is because as the number of total possible worlds increases so does the proportions with zero probabilities. By looking at nature of algorithm 2 we can see that the pruning rate is a polynomially exponential function. Due to this the growth rate for the non pruned worlds in the graph is decreasing.

Finally, we conduct yet another test to verify the time complexity of the algorithm. As seen in the Figure 8 the computation time taken increases at the rate of polynomially exponential function with the increase in the number of pairwise comparisons. With pruning the growth rate of time is much less than without pruning but there is not much significant difference and they both start to grow almost at the same rate once certain limit has crossed. Theoretically, with the decreasing growth rate of non-pruned world the growth rate of computation time should also decrease. The difference exists in this experiment because the computation time shown is not precise enough. It takes a very long time to compute for larger number of pairwise comparisons so we used linear extrapolation to derive the estimated time taken using the computation time taken to solve a small subset of the big problem. Even with pruning the performance of the Possible World Theory in terms of computation time couldn't be improved.

4 CONCLUSION

This work is an attempt to decision making in real world scenario using the concept of Pareto-Optimal objects. The major contribution from this study is in finding confident enough crowd size, incorporating minority opinion of the crowd and decision making using a Slack bot. We have introduced the Possible World Theory, its polynomially

exponential growth curve and some pruning methods to mitigate the issue. The Slack bot developed in the process can be used for decision making within a team. We admit that there are limitations in this work and needs more further revision. The tenacious growth rate of complexity is not so practical and the sample size collected for the purpose of experimentation is also much less than the expected. However, the final ranks obtained provide a much accurate portrayal of the whole crowd behavior. It can be readily used for ranking small set of comparison objects and criteria with higher level of accuracy.

ACKNOWLEDGMENTS

Our sincere appreciation and gratitude to all the scholars from the IDIR lab group who have dedicated their precious time in data collection phase for the experiments and providing invaluable feedbacks.

REFERENCES

- [1] Naeemul Hassan Chengkai Li Gergely V. Zaruba Abolfazl Asudeh, Gensheng Zhang. 2015. Crowdsourcing Pareto-Optimal Object Finding by Pairwise Comparisons. *ACM* 24 (Oct 2015), 753–762. DOI : <https://doi.org/10.1145/2806416.2806451>
- [2] Terrence L Fine Adrianos Papamarcou. 1986. A Note on Undominated Lower Probabilities. *The Annals of Probability* 14 (1986), 981–992.
- [3] Ayoung Suh Christian Wagner. 2014. The Wisdom of Crowds: Impact of Collective Size and Expertise Transfer on Collective Performance. *HICSS* 47 (Jan 2014), 594–603. DOI : <https://doi.org/10.1109/HICSS.2014.80>
- [4] P. B. Sørensen D. B. Lerche. 2008. Evaluation of the ranking probabilities for partial orders based on random linear extensions. *Chemosphere* 53 (Sept 2008), 981–992. DOI : [https://doi.org/10.1016/S0045-6535\(03\)00558-7](https://doi.org/10.1016/S0045-6535(03)00558-7)
- [5] CK Kum H Troidl E Eypasch, R Lefering. 1995. Probability of adverse events that have not yet occurred: a statistical reminder. *British Medical Journal* 311 (1995), 619–620.
- [6] P. Walley. 1991. *Statistical Reasoning with Imprecise Probabilities*. Chapman and Hall, London, Massachusetts.