Optimal Budget Allocation for Crowdsourcing Labels for Graphs (Supplementary Material)

Adithya Kulkarni[1] **Mohna Chakraborty**[2] **Sihong Xie**[3] **Qi Li**[4]

[1,2,4]Computer Science Dept., Iowa State University, Ames, Iowa, USA
[3]Computer Science & Engineering Dept., Lehigh University, Bethlehem, Pennsylvania, USA

A PROOF OF PROPOSITION 2

We use the proof technique proposed by Xie and Frazier [2012] to prove Proposition 2. By Proposition 1, the following value function is obtained:

$$V(S^0) \doteq \sup_{\pi} \mathbb{E}^{\pi}\left[\mathbb{E}\left(\sum_{v\in H_T}\mathbf{1}(v\in H^*)+\sum_{v\notin H_T}\mathbf{1}(v\notin H^*)|\mathcal{F}_T\right)\right] = \sup_{\pi}\mathbb{E}^{\pi}\left(\sum_{v=1}^{N}h(P_v^T(+1))\right). \quad (1)$$

We define $G_0 = \sum_{v=1}^{N} h(P_v^0(+1))$ and $G_{t+1} = \sum_{v=1}^{N} h(P_v^{t+1}(+1)) - \sum_{v=1}^{N} h(P_v^t(+1))$ to decompose the final accuracy $\sum_{v=1}^{N} h(P_v^T(+1))$ into stage-wise reward. Then, $\sum_{v=1}^{N} h(P_v^T(+1))$ can be decomposed as: $\sum_{v=1}^{N} h(P_v^T(+1)) \equiv G_0 + \sum_{t=0}^{T-1} G_{t+1}$. Therefore, the value function can now be re-written as:

$$\begin{aligned} V(S^0) &= G_0(S^0) + \sup_{\pi}\sum_{t=0}^{T-1}\mathbb{E}^{\pi}(G_{t+1}) \\ &= G_0(S^0) + \sup_{\pi}\sum_{t=0}^{T-1}\mathbb{E}^{\pi}(\mathbb{E}(G_{t+1}|\mathcal{F}_t)) \\ &= G_0(S^0) + \sup_{\pi}\sum_{t=0}^{T-1}\mathbb{E}^{\pi}(\mathbb{E}(G_{t+1}|S^t, v_t)). \end{aligned} \quad (2)$$

The first equality holds because G_0 is determinant and independent of the policy π, the second equality holds because of the tower property of conditional expectation and the third equality is true because G_{t+1} depends on $\mathcal{F}_t$ only through S^t and v_t. We define the stage-wise expected reward gained by obtaining the label for the v_t-th instance at the state S^t as:

$$\begin{aligned} R(S^t, v_t) &= \mathbb{E}(G_{t+1}|S^t, v_t) \\ &= \mathbb{E}\left(\sum_{v=1}^{N} h(P_v^{t+1}(+1)) - \sum_{v=1}^{N} h(P_v^t(+1))|S^t, v_t\right). \end{aligned} \quad (3)$$

Therefore, the value function takes the following form:

$$V(S^0) = G_0(S^0) + \sup_{\pi}\mathbb{E}^{\pi}\left(\sum_{t=0}^{T-1} R(S^t, v_t|S^0)\right). \quad (4)$$

B PROOF OF THEOREM 1

To prove the theorem, we first elaborate on the process of belief propagation. Let us take a simple factor graph $FG = (V \cup F, E')$ which is a path graph. Let the path be $v_1 - f_1 - v_2 - f_2 - v_3$. f_1, f_2 represent the pairwise vertex dependency between vertices v_1, v_2 and v_2, v_3, respectively. The message from vertex v_3 to f_2 is initialized with the posterior probability of v_3 (ω_{v_3}). At current timestamp, let the chosen vertex be v_1. Therefore, the messages are propagated from v_3 to v_1 as part of *forward propagation*. For simplicity, let us consider the label to be $+1$. The message from vertex v_3 to factor f_2 is

$$\underset{v_3\to f_2}{\mu}(+1) = \omega_{v_3}(+1),$$

Following Eq. (2), the message from factor f_2 to vertex v_2 is

$$\begin{aligned} \underset{f_2\to v_2}{\mu}(+1) &= \sum_{x'_f=+1, x'_{v_2}=+1}\left(\phi_{f_2}(x'_{f_2})\prod_{v^*\in\{v_2,v_3\}\setminus\{v_2\}}\underset{v^*\to f_2}{\mu}(+1)\right) \\ &= \sum_{x'_f=+1, x'_{v_2}=+1}\left(\phi_{f_2}(x'_{f_2})\underset{v_3\to f_2}{\mu}(+1)\right) \\ &= \phi_{f_2}(+1) * \omega_{v_3}(+1). \end{aligned}$$

Similarly, $\mu_{v_2\to f_1} = \phi_{f_2}(+1) * \omega_{v_3}(+1)$ and $\mu_{f_1\to v_1} = \phi_{f_1}(+1) * \phi_{f_2}(+1) * \omega_{v_3}(+1)$. From the *forward propagation* we can observe that the messages propagated are dependent on factor initialization and posterior probability of the start vertex. Similar observation can be made for *backward propagation* too since both process follow same steps. Therefore, we can conclude that the messages are only updated due to factor initialization and posterior probabilities. Since factor initialization is fixed and does not change with timestamp, the messages are updated only due to the change

Accepted for the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023).

in posterior probabilities. From Eq. (3), the marginal probability of each vertex is dependent on its posterior probability and messages from neighbors. Since messages are updated only due to the change in the posterior probabilities, we can conclude that the marginal probability of each vertex is updated only due to its posterior probability and posterior probabilities of leaf vertices in the graph.

Considering any factor graph FG, when a vertex v_t is chosen, the messages are propagated from leaf vertices to the vertex v_t and from vertex v_t back to the leaf vertices. Each leaf vertex and v_t pair is essentially a path graph. Therefore, our conclusion that the marginal probability of each vertex is updated only due to its posterior probability and posterior probabilities of leaf vertices in the graph is valid for any factor graph FG.

B.1 CONSISTENCY OF GRAPHOBA-OPT

To prove the consistency of GraphOBA-OPT, we utilize the observations from Chen et al. [2013]. As per GraphOBA-OPT, in each iteration we choose a vertex v_t such that

$$v_t = \underset{v}{\operatorname{argmax}} \left(R^+(S^t, v_t) \doteq \max(R_1(S^t, v_t), R_2(S^t, v_t)) \right). \tag{5}$$

Let us consider the computation of expected reward $R^+(S^t, v_t)$. Since the update to posterior probability of each vertex $v \in V$ only occurs due to the obtained label. At a given timestamp, when computing $R_1(S^t, v_t)$ or $R_2(S^t, v_t)$, only the posterior probability of vertex v_t changes. Therefore, the value of the reward only depends on the effect of this change on the graph. As per Eq. (10), the reward is the change in the sum of marginal probabilities in the graph

$$R(S^t, v_t) = \mathbb{E}(\sum_{v=1}^{N} h(P_v^{t+1}(+1)) - \sum_{v=1}^{N} h(P_v^t(+1)) | S^t, v_t). \tag{6}$$

To compute the change in the sum of marginal probabilities in the graph, we first compute the change in marginal probability of vertex v at timestamp t following Chen et al. [2013]. We have

$$P_v^t(x_v) = \omega_v^t(x_v) \prod_{j \in \mathcal{N}(v)} \underset{j \to v}{\mu}(x_v).$$

Therefore,

$$h(P_v^{t+1}(x_v)) - h(P_v^t(x_v)) = (h(\omega_v^{t+1}(x_v) \prod_{j \in \mathcal{N}(v)} \mu_{j \to v}^{t+1}(x_v)) - h(\omega_v^t(x_v) \prod_{j \in \mathcal{N}(v)} \underset{j \to v}{\mu^t}(x_v))).$$

Since the messages from neighbors do not change between two timestamps if v is chosen at timestamp t for obtaining the label. Therefore, $h(P_v^{t+1}(x_v)) - h(P_v^t(x_v)) > 0$ only if $h(\omega_v^{t+1}(x_v)) - h(\omega_v^t(x_v)) > 0$.

Considering any vertex $v' \in \{V - v\}$, if the vertex is not reachable from v then $h(P_{v'}^{t+1}(x_v)) - h(P_{v'}^t(x_v)) = 0$. If it is reachable from v (for simplicity let $v' \in \mathcal{N}(v)$ but the steps are valid even if $v' \notin \mathcal{N}(v)$), then

$$h(P_{v'}^{t+1}(x_{v'})) - h(P_{v'}^t(x_{v'})) = (h(\omega_{v'}^t(x_{v'}) \prod_{j \in \mathcal{N}(v')} \mu_{j \to v'}^{t+1}(x_{v'})) - h(\omega_{v'}^t(x_{v'}) \prod_{j \in \mathcal{N}(v')} \underset{j \to v'}{\mu^t}(x_{v'}))).$$

since $\omega_{v'}^t(x_v)$ does not change between timestamps t and $t+1$. Now considering the messages from neighbors, the messages from all the neighbors except from the factor f' that connects v to v' do not change between timestamps t and $t+1$. Therefore,

$$P_{v'}^{t+1}(x_v) - P_{v'}^t(x_v) \propto \underset{f' \to v'}{\mu^{t+1}}(x_{v'}) - \underset{f' \to v'}{\mu^t}(x_{v'}).$$

However, $\underset{f' \to v'}{\mu}$ is proportional to posterior of v. So

$$h(P_{v'}^{t+1}(x_v)) - h(P_{v'}^t(x_v)) \propto h(\omega_v^{t+1}(x_v)) - h(\omega_v^t(x_v)) \propto \omega_v^{t+1}(x_v) - \omega_v^t(x_v).$$

Since $\forall v' \in \{V - v\}$, the change in marginal probability is either 0 or proportional to $h(\omega_v^{t+1}(x_v)) - h(\omega_v^t(x_v))$. We have that $h(P_{v'}^{t+1}(x_v)) - h(P_{v'}^t(x_v)) > 0$ only if $h(\omega_v^{t+1}(x_v)) - h(\omega_v^t(x_v)) > 0$. Therefore, $\sum_{v=1}^{N} h(P_v^{t+1}(+1)) - \sum_{v=1}^{N} h(P_v^t(+1)) > 0$ only if $h(\omega_v^{t+1}(+1)) - h(\omega_v^t(+1)) > 0$ and $\sum_{v=1}^{N} h(P_v^{t+1}(+1)) - \sum_{v=1}^{N} h(P_v^t(+1)) < 0$ if $h(\omega_v^{t+1}(+1)) - h(\omega_v^t(+1)) < 0$.

The calculation of $h(\omega_v^{t+1}(x_v)) - h(\omega_v^t(x_v))$ is the same posterior calculation as in Chen et al. [2013]. Following the same proof of Chen et al. [2013], we have $\lim_{a_v^t + b_v^t \to \infty} h(\omega_v^{t+1}(x_v)) - h(\omega_v^t(x_v)) = 0$. Therefore, for any $v \in V$, $\lim_{a_v^t + b_v^t \to \infty} h(P_v^{t+1}(+1)) - h(P_v^t(+1)) = 0$. Therefore, $\lim_{a_v^t + b_v^t \to \infty} R(S^t, v_t) = 0$, and thus $\lim_{a_v^t + b_v^t \to \infty} R^+(S^t, v_t) = 0$. Applying other observations from Chen et al. [2013], we have that in any sample path $(v_0, y_{v_0}, ..., v_{t-1}, y_{v_{t-1}})$, GraphOBA-OPT will label each instance infinitely many times as T goes to infinity. Due to our consideration that workers are reliable, if we label each vertex infinitely many times, we will converge to θ_v for each $v \in V$. Therefore, the accuracy will be 100% almost surely implying that GraphOBA-OPT is a consistent policy.

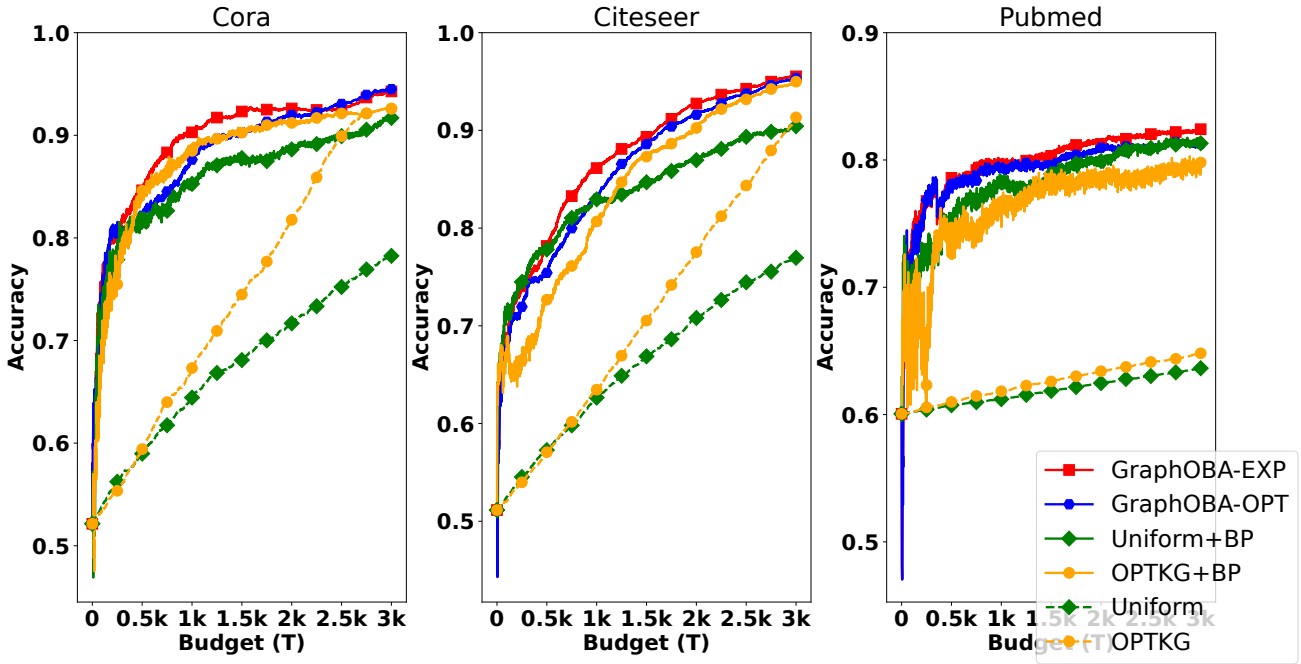

Figure 1: Performance comparison on datasets that follow homophily setting. The plots show the performance on the entire datasets.

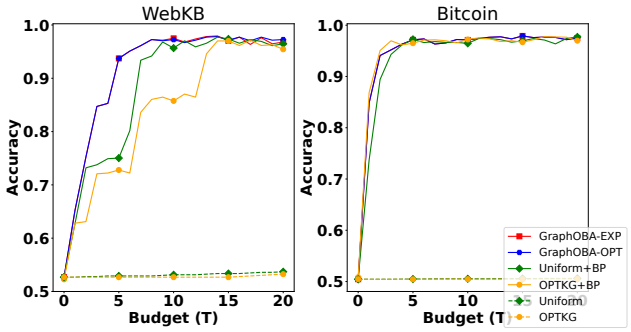

Figure 2: Performance comparison on WebKB and Bitcoin datasets. The plots show the performance on the entire datasets.

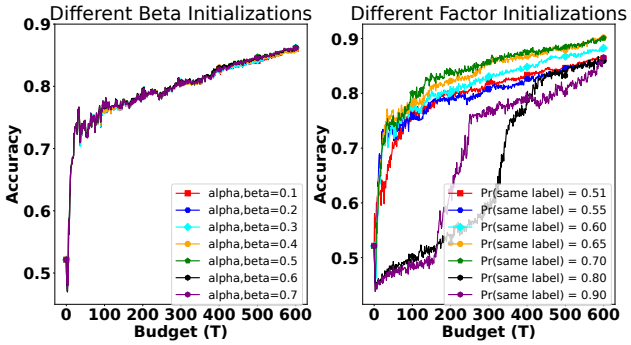

Figure 3: Ablation study results of experiments with different beta and factor initialization on the Cora dataset. We report the performance of GraphOBA-EXP.

B.2 CONSISTENCY OF GRAPHOBA-EXP

As per GraphOBA-EXP, in each iteration we choose a vertex v_t such that

$$R(S^t, v_t) = p_1 * R_1(S^t, v_t) + p_2 * R_2(S^t, v_t). \quad (7)$$

As part of the proof in Section B.1, we show that $R_1(S^t, v_t) > 0$ and $R_2(S^t, v_t) < 0$ when $a_v^t \geq b_v^t + 1$, $R_1(S^t, v_t) < 0$ and $R_2(S^t, v_t) > 0$ when $b_v^t \geq a_v^t + 1$ and $R_1(S^t, v_t), R_2(S^t, v_t) > 0$ when $a_v^t = b_v^t$. Therefore, when $a_v^t = b_v^t$, $R(S^t, v_t) > 0$, but when $a_v^t \neq b_v^t$, $R(S^t, v_t)$ can be 0.

However, even though the change in posterior probability of v_t can be the same when computing R_1 and R_2 especially at the start of execution since all the vertices are initialized with Beta prior distribution $Beta(0.1, 0.1)$, the effect of this change on the graph depends on pairwise vertex dependency initialization in factor vertices. If the probability of vertices v_1 and v_2 having the same label is not equal to 0.5 then $R_1(S^t, v_t) \neq R_2(S^t, v_t)$ and if it is equal to 0.5 then $R_1(S^t, v_t) = R_2(S^t, v_t)$. Therefore, $R(S^t, v_t) \neq 0$ when $a_v^t \neq b_v^t$ if the probability of vertices v_1 and v_2 having the same label is not equal to 0.5. Since we assume that the pairwise vertex dependency among vertices is known, therefore $R(S^t, v_t) \neq 0$. Furthermore, there will be at least one vertex in the graph such that $R(S^t, v_t) > 0$ since all vertices of the graph do not have the same label and pairwise vertex dependency among all pairs of adjacent vertices is

not same.

Since $R(S^t, v_t) > 0$ for any positive integers a_v^t and b_v^t, we can follow the proof technique used in Section B.1 and show that GraphOBA-EXP is a consistent policy.

C ADDITIONAL EXPERIMENTS

We conduct experiments without splitting the dataset into train and test sets. Figure 1 compares our proposed approaches with the baselines on datasets that follow a homophily setting. We provide pseudo code for optimal policy π^* computation for GraphOBA-OPT and GraphOBA-EXP in Algorithm 1. From the results, we can observe that GraphOBA-EXP outperforms the baselines on all three datasets, and GraphOBA-OPT comes second. The results are similar to the results in 1 in the main paper, suggesting that the proposed reward function is efficient and the policies that follow the proposed reward function choose the right vertex to label at each timestamp t. Figure 2 compares GraphOBA-EXP and GraphOBA-OPT with the baselines on WebKB and Bitcoin datasets. The results show that the proposed approaches outperform the baselines for the WebKB dataset and achieve similar performance on the Bitcoin dataset. The results are similar to the results in Figure 2 in the main paper and suggest the importance of knowing dependency among adjacent vertices.

Algorithm 1 Pseudo code for the optimal policy π^* computation for GraphOBA-OPT/GraphOBA-EXP

Input: Unlabeled graph $G = (V, E)$, budget T
Output: Inferred true labels for each vertex $v \in V$
for t in $0...T-1$ **do**
Estimate reward for all vertices of graph G following Eq. (14)/Eq. (15).
Choose the vertex v with the highest estimated reward to request worker label y_{v_t}.
Propagate the labeling information throughout the graph G using belief propagation following Eq. (1) and Eq. (2).
Compute marginal probability of each vertex $v \in V$ following Eq. (3) to infer true label.
return Inferred true label of each vertex $v \in V$.

We observe volatility in performance of OPTKG+BP in Figure 1 (c) for Pubmed dataset, a large dataset with 19717 vertices. Since the budget considered in our experiments is lower than two times the number of vertices in the graph, OPTKG follows a round-robin policy. We conduct addition experiment to understand the reason for the volatility, the results are shown in Table 1. The goal of the experiment is to understand the distribution of vertices chosen by the policies to obtain worker labels. The vertices for which the worker labels are obtained are kept in the set of labeled vertices, and the remaining vertices form the set of unlabeled vertices. For each unlabeled vertex, we find the distance of the nearest labeled vertex and report the mean distance of all vertices.

Table 1: Experiment to understand the reason for volatility in the performance of OPTKG+BP in Figure 1 (c). We report the mean distance of unlabeled vertices to the nearest labeled vertex.

Budget (T)	**OPTKG +BP**	**Uniform +BP**	**GraphOBA-OPT**	**GraphOBA-EXP**
50	4.437	3.713	3.132	3.131
100	3.889	3.400	2.865	2.865
150	3.805	3.103	2.655	2.638
200	3.694	2.988	2.520	2.505
250	3.556	2.863	2.420	2.398
300	3.360	2.771	2.359	2.325
350	3.173	2.702	2.293	2.248
400	3.065	2.631	2.205	2.158
450	2.999	2.552	2.153	2.099
500	2.924	2.511	2.101	2.032

The results in Table 1 show that OPTKG+BP has the largest mean distance compared to other methods. A higher mean distance implies that the labeled vertices are concentrated in a small part of the graph, whereas a lower mean distance implies the labeled vertices are distributed throughout the graph. If labeled vertices are concentrated, when a new worker label is obtained, the newly inferred labels for unlabeled vertices are sensitive to the newly obtained worker label since the new label information is propagated in one direction from the region with more concentrated labeled nodes to the more sparsely labeled regions. Due to this, the labels of all the unlabeled vertices change to the new label, resulting in volatility. If labeled vertices are distributed, the sensitivity for new worker labels is less since the new label information propagates in multiple directions, so that unlabeled nodes can receive information from multiple labeled nodes, resulting in more stability. Furthermore, the volatility decreases as the budget increases since the labeled vertices are no more concentrated in one part of the graph. Figure 1 (c) shows the decrease in volatility with the budget. For the experiments in Figure 1, we do not shuffle the indexes of the vertices, and since OPTKG follows a round-robin policy, it results in the labeled vertices being concentrated in a small part of the graph. The volatility of OPTKG+BP can be reduced by shuffling the indexes of the vertices.

From the discussion in Section 5.5, we observe that the performance of the proposed approach may be sensitive to the initialization of pairwise vertex dependency among adjacent vertices. Therefore, we conduct experiments with different initialization for pairwise vertex dependency and show the results in Figure 3. In the figure, Pr(same label) represents the probability of connect vertices having the same label. From the results, we observe that initializing the pairwise vertex dependency among connect vertices with the probability of both vertices having the same label between 0.65 and 0.7 results in the best performance. The results suggest

that initializing with moderate pairwise vertex dependency among connect vertices is preferred and initializing with very high values can result in a bias towards the label of adjacent vertex, and with very low values can result in over-sensitivity towards the labels provided by the workers.

Furthermore, we conduct experiments with different initialization of α and β and show the results in Figure 3. From the results, we observe that the performance of the proposed approach is not sensitive to the initialization of α and β.

D DATASET PREPROCESSING

None of the five benchmark datasets considered for our experiments are binary-class datasets. Therefore, we first obtain the class distribution in the datasets to convert the datasets from multi-class to binary-class. Then, we combine classes to obtain a nearly equal distribution of vertices. We relabel each vertex with the new binary classes and use the updated datasets for our experiments. Note that we do not assign different labels to the vertices belonging to the same class. Therefore, for some of the datasets, the distribution is not equal.

References

Xi Chen, Qihang Lin, and Dengyong Zhou. Optimistic knowledge gradient policy for optimal budget allocation in crowdsourcing. In *International conference on machine learning*, pages 64–72. PMLR, 2013.

Jing Xie and Peter I Frazier. Sequential bayes-optimal policies for multiple comparions with a control. Technical report, Technical report, Cornell University, 2012.