Task Relevance and Diversity as Worker Motivation in Crowdsourcing

Julien Pilourdault #1, Sihem Amer-Yahia #1, Senjuti Basu Roy *2, Dongwon Lee †3

# *Univ. Grenoble Alpes, CNRS, LIG, France*

1 firstname.lastname@univ-grenoble-alpes.fr

* *NJIT, USA*

2 senjutib@njit.edu

† *Penn State University, USA*

3 dongwon@psu.edu

***Abstract*—Task assignment is a central component in crowdsourcing. Organizational studies have shown that *worker motivation in completing tasks* has a direct impact on the quality of individual contributions. In this work, we examine *motivation-aware task assignment in the presence of a set of workers*. We propose to model motivation as *a balance between task relevance and task diversity* and argue that *an adaptive approach to task assignment* can best capture *the evolving nature of motivation*. Worker motivation is observed and task assignment is revisited appropriately across iterations. We prove the problem to be NP-hard as well as MaxSNP-Hard and develop efficient approximation algorithms with provable guarantees. Our experiments with synthetic data examine the scalability of our algorithms, and our live real data experiments show that capturing motivation using relevance and diversity leads to high crowdwork quality.**

I. Introduction

Task assignment in crowdsourcing is an ongoing research topic [1], [2], [3], [4], [5], [6]. Since the 70's, organizational studies explored worker motivation in physical workplaces [7], and recent work has shown that a similar model prevails in virtual marketplaces [8], [9]. In this paper, we combine these two ideas and investigate motivation-aware task assignment based on the following hypothesis: *Since worker motivation evolves as they complete tasks, tasks should be adaptively re-assigned to them to improve crowdwork quality*. In existing work, motivation is treated as an external factor in task completion, by incentivizing workers for long-lasting tasks [10], [11], or entertaining them when they complete several tasks [12]. In this paper, we take a step toward motivation-aware task assignment and examine its effect on crowdwork quality.

Our first challenge is to model motivation. Some workers may be driven by fun and enjoyment, while others may look to advance their human capital [8], or increase their compensation. Given the low variance in wages of micro-tasks, we propose to focus on two essential factors that matter to workers in choosing tasks: (1) *task diversity* [7] (i.e., how much tasks differ from each other), and (2) *task relevance* (i.e., how much tasks match a worker's qualification). Task diversity has been qualified as a *fun and enjoyment motivation factor* [7]. It can be measured as the distance among the skill requirements of tasks, such as English proficiency for audio transcription. Task relevance, on the other hand, has been largely left to workers (e.g., workers self-appoint themselves to tasks according to their competencies). It can be measured as the distance between the skill requirement of a task and the qualification of a worker. In addition to being natural and essential to capture motivation, both diversity and relevance can be easily computed while workers complete tasks, thereby capturing the evolving nature of motivation. Our objective function combines relevance and diversity to represent motivation as a balance between how different a task from a pool of tasks, and how qualified a worker is for that task.

Our second challenge is to formulate motivation-aware task assignment for a set of workers. This problem was formulated as a one-shot optimization problem whereby goals such as maximizing task quality, or minimizing cost and latency, are used when matching tasks and workers [5], [6]. The difficulty with worker motivation is that *it must be re-captured as workers complete more tasks, and tasks must be assigned to them adaptively.* Therefore, we propose to model task assignment as a series of iterations whereby we observe workers, capture their motivation, and use it to assign new tasks. There exists work on iterative task assignment, but it considers quality [1], [3], [4] or payment [2] as an objective, and not worker motivation Our problem bears similarity to the familiar diversity-relevance trade-off which is known e.g. from IR literature [13], [14]. However, our solutions are very different from solutions to that problem because of the need to serve multiple workers, akin to multiple ranked lists, at once.

We formalize *holistic motivation-aware task assignment*, HTA as a discrete optimization problem with the objective to maximize workers' expected motivation for the tasks assigned to them at a given iteration. We present a rigorous analysis of our problem and prove its non-trivial theoretical results. We show that HTA is NP-Hard using a reduction from the k-partitioning problem KPART [15]. Using an L-reduction [16], we prove that HTA is Max-SNP-Hard and is thus hard to approximate to any arbitrary approximation factor, assessing that there is no polynomial time approximation scheme for HTA, unless $P = NP$.

Our third challenge is to design efficient algorithms to solve HTA. We first develop HTA-APP with a $1/4$-approximation guarantee similar in the spirit of maximum quadratic assignment problem [16]. We realize that one of the subroutines that HTA-APP uses has to solve the *Linear Sum Assignment Problem*(LSAP) [17] using Hungarian algorithm that causes significant computational overhead. We then propose a greedy solution to the LSAP problem, which has $1/2$ approximation factor, yet terminates asymptotically faster than the Hungarian algorithm. We design a new algorithm HTA-GRE, that combines two approximation algorithms. This new algorithm is asymptotically faster than HTA-APP, yet has provable guarantees, as we show that HTA-GRE admits $1/8$ approximation factor to solve HTA.

We run two sets of experiments with real tasks: a simulation with synthetic workers, to examine scalability and the value of the objective function, and a live experiment to study the end-to-end performance of our system. We validate our hypothesis by showing that accounting for diversity and adaptivity provide the best compromise between *outcome quality*, *task throughput,* and *worker retention.*

This paper makes the following contributions:

1) **Problem Formulation and Analysis (Section III).** We formalize worker motivation as a balance between task diversity and relevance, and define the holistic motivation-aware task assignment problem, HTA. We prove that HTA is both NP-hard and MaxSNP-hard.
2) **Algorithms (Section IV).** We develop two approximation algorithms, HTA-APP and HTA-GRE with provable guarantees.
3) **Platform and Validation (Section V).** We run experiments on our newly developed platform and assess the utility of an adaptive approach for obtaining high quality contributions from the crowd for multiple tasks, in reasonable time.

II. PRELIMINARIES

Tasks and Workers. We consider a set of tasks $\mathcal{T} = \{t_1, \ldots, t_N\}$, a set of workers $\mathcal{W} = \{w_1, \ldots, w_Q\}$ and a set of keywords $\mathcal{S} = \{s_1, \ldots, s_R\}$. A task t is represented by a vector $\langle t(s_1), t(s_2), \ldots, t(s_R)\rangle$ where each $t(s_i)$ is a Boolean value that denotes the presence or absence of keyword s_i in task t. A keyword associated to a task reflects its content and requirements. In Amazon Mechanical Turk, for instance, an audio transcription task is often associated with keywords such as *"audio"*, *"English"*, and *"news"*, while a video tagging task is associated with keywords such as *"Google street view"* and *"tagging"*. In CrowdFlower, a sentiment analysis task is often associated to *"sentiment analysis"* and *"English."*

A worker w is a vector $w = \langle w(s_1), \ldots, w(s_R)\rangle$ where each $w(s_i)$ is a Boolean value capturing the *expressed interest* of w in tasks with keyword s_i.

Worker Motivation. We propose to capture motivation *as a balance* between *task diversity*, i.e., how different tasks are from each other, and *task relevance*, i.e., how proficient a worker believes to be for a task. These two factors are easily measurable and can be updated on-the-fly as workers complete tasks. Indeed, several other factors could be used to reflect workers' motivation. The six factors that influence motivation the most are *Payment, Task Autonomy, Task Diversity (a.k.a. Skill Variety), Task Identity, Human Capital Advancement, Pastime* [8]. While some workers look for fun and enjoyment, others want to learn something new, pass time, or make some money. Our focus on micro-tasks makes payment less important since they tend to have comparable rewards (e.g., $<$ \$0.15). Task autonomy and task identity are often minimal since micro-tasks do not yield a high degree of freedom and usually do not allow workers to have a tangible result of their work. Human capital advancement, i.e., choosing a task for its capacity to increase one's knowledge, is also negligible. Finally, picking a task to pass time is difficult to measure. Therefore, we focus on task diversity and relevance in modeling motivation. That will also allow us to isolate their effect from other factors.

Task Diversity. We first define $d(t_k, t_l)$, the *pairwise task diversity* between two tasks t_k and t_l. It essentially measures the aggregated differences of keywords between two tasks. In our setting, we use the Jaccard similarity function J as follows: $d(t_k, t_l) = 1 - J(\langle t_k(s_1), \ldots, t_k(s_R)\rangle, \langle t_l(s_1), \ldots, t_l(s_R)\rangle)$. In practice, we allow $d()$ to be any distance function (e.g., Euclidean, Jaccard) as long as it is a metric, i.e., it verifies the triangular inequality. The diversity $\mathit{TD}(\mathcal{T}')$ of a set of tasks $\mathcal{T}' \subseteq \mathcal{T}$ is captured in the usual manner, by aggregating the pairwise distances in $\mathcal{T}'$:

$$\mathit{TD}(\mathcal{T}') = \sum_{\substack{t_k, t_l \in \mathcal{T}' \\ k > l}} d(t_k, t_l) \tag{1}$$

Task Relevance. Let $\mathit{rel}(t, w)$ be the function that evaluates how relevant a task t is to a worker w: $\mathit{rel}(t, w) = 1 - d_{\mathit{rel}}(\langle t_{s_1}, \ldots, t_{s_R}\rangle, \langle w_{s_1}, \ldots, w_{s_R}\rangle)$. We also use Jaccard to express $d_{\mathit{rel}}()$. The relevance $\mathit{TR}(\mathcal{T}', w)$ of a set of tasks $\mathcal{T}'$ for a worker w is captured by aggregating each pairwise distance for $t \in \mathcal{T}'$.

$$\mathit{TR}(\mathcal{T}', w) = \sum_{t \in \mathcal{T}'} \mathit{rel}(t, w) \tag{2}$$

Finally, we wish to capture the expected motivation of a worker w for a set of tasks $\mathcal{T}' \subseteq \mathcal{T}$ as a simple linear combination of diversity and relevance [1] of tasks in $\mathcal{T}'$:

$$\mathit{motiv}(\mathcal{T}', w) = 2\alpha_w \mathit{TD}(\mathcal{T}') + \beta_w(|\mathcal{T}'| - 1) \times \mathit{TR}(\mathcal{T}', w) \tag{3}$$

where (α_w+β_w=1). These weights capture the preference of a worker w has for those factors. They can be observed and captured for each worker. The other components, 2 and $(|\mathcal{T}'| - 1)$, are normalization factors as in [13].

While we focus on combining diversity and relevance to express motivation, our problem formulation and algorithms (Sections III and IV) are applicable to other motivation factors if expressed as metrics.

III. Problem Formulation

In this work, we want to observe workers as they complete tasks, refine our understanding of their motivation, and *adaptively* re-assign tasks to them. We view adaptive task assignment as a series of iterations. How to define an iteration is orthogonal to our problem. Indeed, an iteration could be defined by discretizing time into equal-sized windows, by the number of available/completed tasks, by the number of workers, or a combination thereof. We will see in Section V how we define an iteration in our experiments.

Task Assignment in Iterations. At iteration i, we assign a set of tasks $\mathcal{T}^i_w \subseteq \mathcal{T}^i$ to workers $w \in \mathcal{W}^i$. Here, $\mathcal{T}^i$ (resp. $\mathcal{W}^i$) refers to all available tasks (resp. workers) at iteration i. We also observe each worker w as she chooses and completes tasks and update her α^i_w and β^i_w. Suppose a worker w completes a task $t_j \in \mathcal{T}^{i-1}_w$, after having completed tasks $\{t_1, \ldots, t_{j-1}\}$. We compute the marginal gain in task diversity $\sum_{t_k \in \{t_1,\ldots,t_{j-1}\}} d(t_j, t_k)$ (resp. task relevance $\mathit{rel}(t_j, w)$) brought by t_j. We normalize that gain by the maximum gain that could be obtained using tasks in $\mathcal{T}^{i-1}_w \backslash \{t_1, \ldots, t_{j-1}\}$. We compute α^i_w (resp. β^i_w) as the average of all collected gains (one for each completed task).

We can now revisit Equation 3 to reflect a worker's motivation per iteration i: $\mathit{motiv}(\mathcal{T}^i_w, w) = 2\alpha^i_w \mathit{TD}(\mathcal{T}^i_w) + \beta^i_w(|\mathcal{T}^i_w| - 1) \times \mathit{TR}(\mathcal{T}^i_w, w)$. Then, we study the following problem:

Problem 1 (Holistic Task Assignment — HTA)**.** For a set of workers $\mathcal{W}^i \subseteq \mathcal{W}$ available at iteration i, choose $\mathcal{T}^i_w \subseteq \mathcal{T}^i$, one for each worker $w \in \mathcal{W}^i$, such that:

$$\begin{aligned} \textbf{arg max} \quad & \sum_{w \in \mathcal{W}^i} \mathit{motiv}(\mathcal{T}^i_w, w) \\ & \forall w \in \mathcal{W}^i,\ |\mathcal{T}^i_w| \leq X_{max} \qquad (C_1) \\ & \forall w, w' \in \mathcal{W}^i,\ \mathcal{T}^i_w \cap \mathcal{T}^i_{w'} = \emptyset \quad (C_2) \end{aligned}$$

X_{max} in constraint C_1 limits the number of tasks assigned to each worker and prevents task overload. C_2 guarantees that each task is assigned to at most one worker. To ensure adaptive task assignment, we solve HTA at each iteration. Once assigned, a task is dropped from subsequent iterations.

[1] The linear combination allows us to design efficient algorithms with theoretical guarantees.

A. NP-Completeness

HTA is a difficult problem since it relates to well-known NP-Hard problems. We bring here the formal proof of NP-completeness.

Theorem 1. *The decision version of* HTA *is NP-complete.*

Proof: The decision version of HTA is as follows: *Instance*: tasks $\mathcal{T}^i$, workers $\mathcal{W}^i$, and their (α^i_w, β^i_w), X_{max} and an objective value Z; and *Question*: are there $|\mathcal{W}^i|$ disjoint sets, $\mathcal{T}^i_w \subseteq \mathcal{T}^i$, for each worker w, of size X_{max}, such that $\sum_{w \in \mathcal{W}^i} \mathit{motiv}(\mathcal{T}^i_w, w) \geq Z$?

(1) HTA $\in NP$: Since a nondeterministic algorithm needs only to guess $|\mathcal{W}^i|$ sets and can verify the question in polynomial time, HTA $\in NP$.

(2) HTA is NP-hard : Let us consider the k-partitioning problem KPART [15]. The decision version of this problem is as follows. *Instance*: a weighted graph $G = (V, E, \omega)$, integers δ and k s.t. $\delta \geq 3$ and $|V| = k \times \delta$, an objective value Y; and *Question*: is there a partition of V into k disjoint sets $V_1, \ldots, V_k$ with $|V_l| = \delta, \forall l \in [\![1, k]\!]$ s.t. $\sum_{l \in [\![1,k]\!]} \sum_{u,v \in V_l} \omega(\{u, v\}) \geq Y$? KPART is known to be NP-complete [15] using a reduction from graph partitioning [18]. The reduction works as follows. Each vertex $v \in V$ is mapped to a task $t \in \mathcal{T}^i$. The weight of an edge $(u, v) \in E$ is mapped to skill diversity between two tasks: $\omega(\{u, v\}) = 2 * d(t_u, t_v)$. If there is no edge $\{u, v\} \in E$, we set $d(t_u, t_v) = 0$. We consider $|\mathcal{W}^i| = k$ highly-skilled workers who prefer skill diversity by setting for each $w \in \mathcal{W}^i$ $\alpha^i_w = 1, \beta^i_w = 0$. We set $X_{max} = \delta$ and $Z = Y$. An instance of KPART thus defines an instance of HTA where k workers receive δ tasks and where the objective function is simplified to $2 * \sum_{w \in \mathcal{W}^i} \sum_{t,t \in \mathcal{T}^i_w} d(t, t')$. Note that in such an instance $k \times \delta = |\mathcal{T}^i|$, therefore, all tasks are assigned. This transformation can be done in polynomial time. KPART has a solution if and only if this instance of HTA has a solution. This proves NP-Completeness. ■

B. Max-SNP-Hardness

We show that in addition to being NP-complete, HTA does not admit a polynomial time approximation scheme. HTA is related to well-known problems that cannot be approximated in some cases. In particular, HTA is similar to the *Maximum Quadratic Assignment Problem* (MAXQAP), which is known to be Max-SNP-Hard [16].

Theorem 2. HTA *is Max-SNP-Hard.*

Proof: (Sketch) To prove Max-SNP-Hardness, we use an L-reduction (linear reduction) using the maximum quadratic assignment problem (MAXQAP), which is Max-SNP-Hard [16]. An L-reduction in this case is a polynomial-time mapping taking each instance I of MAXQAP to an instance I' of HTA such that there are positive constants a and b that make the following statements true:

- (L1) $OPT(I') \leq a \times OPT(I)$
- (L2) For any solution of I' with objective function value OBJ', we can, in polynomial time, find a solution of I with value OBJ such that $(OPT(I) - OBJ)$ is no more than $b \times (OPT(I') - OBJ')$

We describe the maximum quadratic assignment problem (MAXQAP). We adopt the formulation of Arkin et al. [16]. In MAXQAP, three $M \times M$ non-negative symmetric matrices $A = (a_{k,l}), B = (b_{k,l}), C = (c_{k,l})$ are given, and the objective is to compute a permutation $\pi()$ of $\mathcal{Y} = \{1, \ldots, M\}$ that maximizes: $\sum_{k,l \in \mathcal{Y},\ k \neq l} a_{\pi(k),\pi(l)} b_{k,l} + \sum_{k \in \mathcal{Y}} c_{k,\pi(k)}$. Intuitively, when A and B are the adjacency matrices of two graphs, this formulation of MAXQAP is equivalent to finding a subgraph in B that is isomorphic to A such that the total weight is maximized. We now consider an instance I of MAXQAP where matrices A, B and C are defined as follows. We set A as the $M \times M$ adjacency matrix of Y cliques of size X and $M - Y * X$ isolated vertices. All edges of the same clique are labeled with the same weight. We set B as the $M \times M$ adjacency matrix of a complete graph with M vertices and edges labeled with a weight satisfying the triangular inequality. We set C as a $M \times M$ matrix with the same values in each group of X consecutive columns up to the $M - Y * X$-th column. All other values are 0. We suppose that weights A and B are not zero and $Z \geq 3$. Now, we transform this MAXQAP instance I to an instance I' of HTA using the following steps: (1) We set $X_{max} = X$; (2) We set $|\mathcal{T}^i| = M$ tasks that satisfy $\forall k, l \in 1, \ldots, |\mathcal{T}^i|\ d(t_k, t_l) = b_{k,l}$; (3) We set $|\mathcal{W}^i| = Y$ workers and $\forall q \in 1, \ldots, |\mathcal{W}^i|\ \alpha^i_{w_q} = a_{X_{max}(q-1)+1, X_{max}(q-1)+1}, \beta^i_{w_q} = c_{1, X_{max}*(q-1)+1}/(X_{max} - 1)$ and, $\forall k \in 1, \ldots, |\mathcal{T}^i|\ rel(w_q, t_k) = 1$; (4) We set $\mathcal{T}^i_{w_q} = \{t_k \mid \lceil \pi(k)/X_{max} \rceil = q\}$. Clearly, such a transformation is done in polynomial time. The following lemmas substantiate the proof.

Lemma 1. *$OPT(I') = a \times OPT(I)$, where $a = 1$.*

Lemma 2. *$(OPT(I) - OBJ) = b \times (OPT(I') - OBJ')$, where $b = 1$.*

We omit the proofs of both lemmas for brevity and refer to Section IV-A where we show that the value of the objective function of these two seemingly different problems are actually identical (Equation 8). Therefore, given an instance of I of HTA, its optimum objective function value is the same as that of I' of MAXQAP. In addition, the gap in the objective function value between the optimum and any other solution is the same for both problems. From these two aforementioned lemmas, we verify the conditions specified in (L1) and (L2) above that are necessary for the proof. ■

IV. TASK ASSIGNMENT ALGORITHMS

Since HTA is NP-hard and even hard to approximate, it is prohibitively expensive to solve on large instances. In our scenario, response time is important since task assignment has to be solved *online*, at each iteration i. The challenge is to design efficient algorithms that also have provable approximation factors. We propose two algorithms. They both rely on the assumption that the distance function used to model diversity is a metric. That is not an overstretch as Jaccard is indeed a metric [19]. When that property is not assumed, HTA remains largely inapproximable [16], [17], [20]. Our first algorithm has a better approximation factor but incurs a higher running time. The second one compromises on the approximation factor to ensure a faster running time.

$rel(t,w)$	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8
w_1	0.28	0.25	0.2	0.43	0.67	0.4	0	0.4
w_2	0.3	0	0.2	0.25	0.25	0	0	0.4

TABLE I
EXAMPLE OF TASKS AND WORKERS

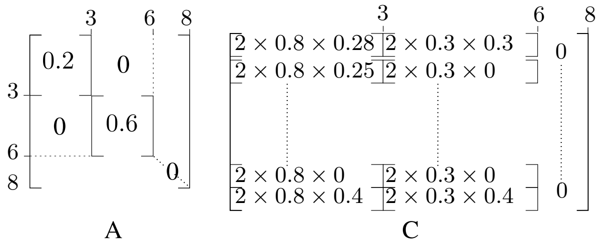


Fig. 1. Mapping to MAXQAP - Matrices A and C

A. MAXQAP *and* HTA

We now show that the objective functions of MAXQAP and HTA are actually identical. We first describe the maximum quadratic assignment problem (MAXQAP) using the formulation of Arkin et. al[16]. In MAXQAP, three $M \times M$ non-negative symmetric matrices $A = (a_{k,l}), B = (b_{k,l}), C = (c_{k,l})$ are given, and the objective is to compute a permutation $\pi()$ of $\mathcal{Y} = \{1, \ldots, M\}$ that maximizes:

$$\sum_{k,l \in \mathcal{Y},\ k \neq l} a_{\pi(k),\pi(l)} b_{k,l} + \sum_{k \in \mathcal{Y}} c_{k,\pi(k)}$$

MAXQAP degenerates to a special case when A and B are adjacency matrices of graphs [16], [17], [20].

Recall that HTA admits tasks $\mathcal{T}^i = \{t_1, \ldots, t_{|\mathcal{T}^i|}\}$ and workers $\mathcal{W}^i = \{w_1, \ldots, w_{|\mathcal{W}^i|}\}$. First, we define A to be the $|\mathcal{T}^i| \times |\mathcal{T}^i|$ adjacency matrix of $|\mathcal{W}^i|$ disjoint cliques of X_{max} vertices and $|\mathcal{T}^i| - |\mathcal{W}^i| * X_{max}$ isolated vertices. Each clique corresponds to a worker. In each clique, edges are labeled with α^i_w. Thus, in the graph whose adjacency matrix is A, each worker is mapped to a set of vertices. We set B to be the adjacency matrix of the complete graph where vertices are mapped to tasks in $\mathcal{T}^i$ and edges are labeled with pairwise task diversities. We define the $|\mathcal{T}^i| \times |\mathcal{T}^i|$ matrix C to represent the linear part of our objective function, i.e., the relevance part. If l is a column that corresponds to a worker w in matrix A, then $c_{k,l}$ equals $rel(w, t_k)$ multiplied by $\beta^i_w * (X_{max} - 1)$. Formally:

$$\forall k,l \in 1,\ldots,|\mathcal{T}^i|,\ a_{k,l} = \begin{cases} \alpha^i_{w_{\lceil l/X_{max}\rceil}} & \textit{if } \lceil l/X_{max}\rceil \le |\mathcal{W}^i| \wedge \lceil k/X_{max}\rceil = \lceil l/X_{max}\rceil \\ 0 & \textit{else} \end{cases} \quad (4)$$

$$\forall k,l \in 1,\ldots,|\mathcal{T}^i|,\ b_{k,l} = d(t_k,t_l) \quad (5)$$

$$\forall k,l \in 1,\ldots,|\mathcal{T}^i|,\ c_{k,l} = \begin{cases} \beta^i_{w_{\lceil l/X_{max}\rceil}} \textit{rel}(w_{\lceil l/X_{max}\rceil},t_k)(X_{max}-1) & \textit{if } l \le |\mathcal{T}^i| - |\mathcal{W}^i| * X_{max} \\ 0 & \textit{else} \end{cases} \quad (6)$$

Example 1. Suppose that we have 2 workers and 8 tasks as exposed in Table I. Let $X_{max} = 3$, $\alpha^i_{w_1} = 0.2$, $\beta^i_{w_1} = 0.8$, $\alpha^i_{w_2} = 0.6$ and $\beta^i_{w_1} = 0.3$. Figure 1 shows matrices A and C, as defined by Equations 4 and 6. The matrices A and C are $|\mathcal{T}^i| \times |\mathcal{T}^i| = 8 \times 8$ matrices where lines and columns are mapped to tasks. In A, each square sub-matrix of size $X_{max} \times X_{max} = 3 \times 3$ corresponds to a given worker (Equation 4). For instance, the first sub-matrix corresponds to worker w_1 where $\alpha^i_{w_1} = 0.2$. In C, each group of $X_{max} = 3$ consecutive columns corresponds to a given worker. Each line corresponds to the relevance gain induced by the task on this line (Equation 6). For instance, $c_{1,1} = (X_{max} - 1) * \beta^i_{w_1} * \textit{rel}(w_1,t_1) = 2 \times 0.8 \times 0.28$.

Given that A and B are adjacency matrices, solving our MaxQap instance is equivalent to finding a subgraph in B that is isomorphic to A such that the objective function is maximized [16]. Indeed, a permutation π of $\{1,\ldots,|\mathcal{T}^i|\}$ maps each vertex of B to a vertex of A. In our settings, this maps a task to a vertex in A, which may be associated to a worker. Then, let us consider a pair of tasks (t_k,t_l) that is assigned to the pair of vertices $(\pi(k),\pi(l))$. The profit induced by this pair is $a_{\pi(k),\pi(l)} * b_{k,l} + c_{k,\pi(k)} + c_{l,\pi(l)}$. Thus, if a worker w_q is associated to both vertices $\pi(k)$ and $\pi(l)$ in A, the profit induced by this pair is:

$$\alpha^i_{w_q} * d(t_k,t_l) + \beta^i_{w_q}(X_{max}-1)(\textit{rel}(w_q,t_k) + \textit{rel}(w_q,t_l))$$

The left part $\alpha^i_{w_q} * d(t_k,t_l)$ corresponds to task diversity. Since A is composed of $|\mathcal{W}^i|$ cliques and given the objective function, $\alpha^i_{w_q} * d(t_k,t_l)$ is counted for each pair of tasks that is associated to the same worker (*if* clause in Equation 4). The right part $\beta^i_{w_q}(X_{max}-1)(\textit{rel}(w_q,t_k)+\textit{rel}(w_q,t_l))$ corresponds to task relevance. It is counted for each task that is associated to a worker (*if* clause in Equation 6).

Now, we can map the output of the MaxQap instance to the output of Hta. For all workers w_q in $\mathcal{W}^i$, we set:

$$\mathcal{T}^i_{w_q} = \{t_k \mid \lceil \pi(k)/X_{max}\rceil = q\} \quad (7)$$

Equation 7 partitions $\mathcal{T}^i$ into $|\mathcal{W}^i|$ sets using the solution π of the MaxQap instance.

Algorithm 1 Hta-App

```
Input: $\mathcal{W}^i = \{w_1,\ldots,w_{|\mathcal{W}^i|}\}$, $\mathcal{T}^i = \{t_1,\ldots,t_{|\mathcal{T}^i|}\}$
Output: $|\mathcal{W}^i|$ sets of tasks $\{\mathcal{T}^i_{w_1},\ldots,\mathcal{T}^i_{w_{|\mathcal{W}^i|}}\}$
 1: Build matrices $A, B, C$ (Equations 4, 5 and 6)
 2: $M_B \leftarrow$ maximum weight matching in $B$
 3: for $k \in 1,\ldots,|\mathcal{T}^i|$
 4:     $deg^A_k = \sum_{l \in 1,\ldots,|\mathcal{T}^i| \setminus \{k\}} a_{k,l}$
 5:     if $t_k$ is covered by $M_B$
 6:         $b_M(t_k)$ = weight of the edge in $M_B$ incident to $t_k$
 7:     else
 8:         $b_M(t_k) = 0$
 9: for $k,l \in 1,\ldots,|\mathcal{T}^i|$
10:     $f_{k,l} = b_M(t_k) deg^A_l + c_{k,l}$
11: $\pi' \leftarrow$ an optimal solution to the linear assignment problem $\max_\sigma \sum_k f_{k,\sigma_k}$
12: for $(t_k,t_l) \in M_B$
13:     $\pi(k) = \pi'(k)$ and $\pi(l) = \pi'(l)$ with probability $1/2$
14:     $\pi(k) = \pi'(l)$ and $\pi(l) = \pi'(k)$ otherwise
15: if $k \notin M_B$
16:     $\pi(k) = \pi'(k)$
17: for $w_q \in \mathcal{W}^i$
18:     $\mathcal{T}^i_{w_q} \leftarrow \{t_k \mid \lceil \pi(k)/X_{max}\rceil = q\}$
19: return $\{\mathcal{T}^i_{w_1},\ldots,\mathcal{T}^i_{w_{|\mathcal{W}^i|}}\}$
```

Example 2. We continue with Example 1. Suppose that the solution of the MaxQap instance returns $\pi(1) = 4, \pi(4) = 1$ and $\forall k \in 1,\ldots,|\mathcal{T}^i|, k \notin \{1,4\}\ \pi(k) = k$. Given Equation 7, we have $\mathcal{T}^i_{w_1} = \{t_4,t_2,t_3\}$ and $\mathcal{T}^i_{w_2} = \{t_1,t_5,t_6\}$. Tasks t_7 and t_8 are left unassigned.

Following the previous observations, we can show that:

$$\sum_{w \in \mathcal{W}^i} \textit{motiv}(\mathcal{T}^i_w, w) = \sum_{\substack{k,l \in 1,\ldots,|\mathcal{T}^i| \\ k \ne l}} a_{\pi(k),\pi(l)} b_{k,l} + \sum_{k \in 1,\ldots,|\mathcal{T}^i|} c_{k,\pi(k)} \quad (8)$$

This shows that the value of the objective function of our Hta instance is same as the MaxQap instance that we built. We omit the proof for brevity.

B. *Approximation Algorithm* Hta-App

In Section IV-A, we showed how we can map an instance of Hta to an instance of MaxQap. We now present Hta-App (Algorithm 1). First Hta-App maps tasks and workers to matrices as exposed in Section III-B (Line 1 in Algorithm 1). Matrix A is the weighted adjacency matrix of $|\mathcal{W}^i|$ cliques (one per worker). Matrix B is the weighted adjacency matrix of the graph where vertices are mapped to tasks and edges are labeled with their pairwise task diversity. Matrix C represents the task relevance part, where each row represents a task and each column a possible assignment for this task. Then, from

Lines 2 to 16, HTA-APP uses only these matrices. First, it finds a maximum weight matching M_B with respect to task diversity using matrix B. This step identifies a set of task pairs whose aggregated diversity value is maximized. Then, HTA-APP builds an auxiliary problem using M_B and matrices A and C. This problem is an instance of the *Linear Sum Assignment Problem*(LSAP) [17]. In Line 10, HTA-APP combines the profit associated to task diversity and task relevance. Observe that if a vertex k is not associated to a worker in matrix A, then $\forall k \in 1, \ldots, |\mathcal{T}^i|$ $f_{k,l} = 0$. Conversely, if k is associated to worker w_q and if task t_k is incident to the edge $(t_k, t_l) \in M_B$, we have $f_{k,l} = d(t_k, t_l) * (X_{max} - 1) * \alpha^i_{w_q} + \beta^i_{w_q} * (X_{max} - 1) * rel(w_q, t_k)$. The rationale behind using this auxiliary problem with those weights is to build a *linear* problem that is easier to solve since our original problem is inherently *quadratic*. Intuitively, $f_{k,l}$ is an approximated profit obtained when assigning task t_k to vertex l. In Line 11, HTA-APP solves the LSAP instance. Then, it uses a slightly modified version of the solution π' of the LSAP problem. For each edge (t_k, t_l) in the maximum matching M_B, it randomly permutes the assignation of t_k and t_l (Lines 12-14). If a task is not in the matching, it is not permuted (Lines 15-16) further. Finally, HTA-APP builds a solution for HTA using Equation 7 (Lines 17-18). Note that HTA-APP is an adaptation of the algorithm of Arkin et al. [16] that was designed for MAXQAP.

Example 3. We run HTA-APP using workers and tasks from Example 1. Suppose that $M_B = \{(t_4, t_8), (t_1, t_6), (t_3, t_2), (t_7, t_5)\}$ and $d(t_4, t_8) = 1, d(t_1, t_6) = 1, d(t_3, t_2) = 0.86, d(t_7, t_5) = 0.8$ (Algorithm 1 - Line 2). Then, HTA-APP will set $f_{1,1} = 1 * 0.4 + 0.448 = 0.848$. This profit is used in the LSAP instance in Line 11. It is an approximation of the profit obtained when assigning task t_1 to vertex 1, which is associated to worker w. Then, HTA-APP solves LSAP and we obtain $\pi = (4, 7, 1, 6, 3, 8, 2, 5)$ (Line 16). Therefore, worker w_1 receives tasks t_3, t_5, t_7 and worker w_2 tasks t_1, t_4, t_8.

Theorem 3. HTA-APP *is a $\frac{1}{4}$-approximation algorithm for the* HTA *problem.*

Proof: (Sketch): We use an approximation-preserving reduction to prove the approximation factor [21]. In Section IV-A, we exposed how to map any instance of HTA to an instance of MAXQAP. We also showed that an optimal solution for the obtained instance of MAXQAP is an optimal solution for HTA. With this non-trivial one to one mapping between HTA and MAXQAP, a non-trivial approximation algorithm for MAXQAP will also solve HTA with the same approximation guarantee. The rest of the proof is akin to [16] (Algorithm 1 - Lines 2-16), which has a $\frac{1}{4}$-approximation factor for the MAXQAP problem. Therefore, HTA-APP is also a $\frac{1}{4}$-approximation for HTA. ■

Lemma 3. HTA-APP *runs in* $\mathcal{O}(|\mathcal{T}^i|^3)$ *time*

Proof: The cubic time complexity comes from the linear assignment problem that the algorithm needs to solve (Algorithm 1 - Line 11). This kind of problem is typically solved using the Hungarian algorithm that runs in $\mathcal{O}(|\mathcal{T}^i|^3)$ time [17]. The matching step (Line 2) can be done in $\mathcal{O}(|\mathcal{T}^i|^2 \log |\mathcal{T}^i|)$ time using a greedy matching [16]. ■

Algorithm 2 HTA-GRE

```
Input: $\mathcal{W}^i = \{w_1, \ldots, w_{|\mathcal{W}^i|}\}$, $\mathcal{T}^i = \{t_1, \ldots, t_{|\mathcal{T}^i|}\}$
Output: $|\mathcal{W}^i|$ sets of tasks $\{\mathcal{T}^i_{w_1}, \ldots, \mathcal{T}^i_{w_{|\mathcal{W}^i|}}\}$
 1: Build matrices $A, B, C$ (Equations 4, 5 and 6)
 2: $M_B \leftarrow$ maximum weight matching in $B$
 3: ... (Lines 3-10 of Algorithm 1)
11: $\pi' \leftarrow$ greedy matching in the completed bipartite graph
    associated with the LSAP problem $\max_\sigma \sum_k f_{k,\sigma_k}$
12: ... (Lines 12-18 of Algorithm 1)
19: return $\{\mathcal{T}^i_{w_1}, \ldots, \mathcal{T}^i_{w_{|\mathcal{W}^i|}}\}$
```

C. *Approximation Algorithm* HTA-GRE

Although HTA-APP tackles the complexity of our problem by returning, in polynomial time, a solution that verifies a performance guarantee, its running time may not be satisfactory. We therefore propose an alternative algorithm HTA-GRE that runs in $\mathcal{O}(|\mathcal{T}^i|^2 \log |\mathcal{T}^i|)$ time, where the approximation factor is slightly compromised.

In Lemma 3, we observed that the time complexity of HTA-APP is dominated by the time complexity of the algorithm that solves the LSAP problem (Algorithm 2 - Line 11). This step can be done in $\mathcal{O}(|\mathcal{T}^i|^3)$ time, using improved versions of the well-known Hungarian algorithm [17]. To the best of our knowledge $\mathcal{O}(|\mathcal{T}^i|^3)$ is the best *polynomial* time complexity we can obtain to solve this assignment step. There also exists a number of algorithms that solve LSAP in *pseudo-polynomial* time [17] (e.g. *cost-scaling* algorithms that run in $\mathcal{O}(|\mathcal{T}^i|^{\frac{5}{2}} \log(|\mathcal{T}^i|\mathcal{C}))$ time where $\mathcal{C} = \max_{k,l \in 1,\ldots,|\mathcal{T}^i|} f_{k,l}$). Since our goal is to reach a *polynomial* time approximation, those algorithms are not applicable.

Our proposed HTA-GRE algorithm (Algorithm 2) improves the cubic running time complexity. Our rationale is the following: rather than solving the LSAP problem *optimally*, we aim to find an *approximate* solution in polynomial time (Algorithm 2-Line 11). Specifically, we find a greedy matching in the complete weighted bipartite graph associated to the LSAP instance. The rest of the algorithm remains unchanged. In the next paragraph, we describe how we find this approximation. Then, we show that HTA-GRE is a $\frac{1}{8}$-approximation algorithm that runs in $\mathcal{O}(|\mathcal{T}^i|^2 \log |\mathcal{T}^i|)$ time. Our novelty is that our final solution still admits an approximation factor.

Approximation for LSAP*:* We present an approximation algorithm to solve LSAP. For that, we first redefine the LSAP problem using graph theory [17]. LSAP can be modeled using a weighted complete bipartite graph $\mathcal{G}^{\text{LSAP}} = (U, V, E, f)$. We set $U = V = \{1, \ldots, |\mathcal{T}^i|\}$. Each edge $e = (k, l)$, $k \in$

$U, l \in V$ has the weight $f_{k,l}$. Solving LSAP becomes equivalent to solving a *Maximum Weight Perfect Matching Problem* (Mwpmp) [17], [22]. In graph theory, the problem of finding a matching, i.e., a set of vertex-disjoint edges, with the maximal weight is called the *maximum weight matching problem* (Mwmp) [22]. A *maximum weight perfect matching* is a maximum weight matching that covers all vertices. To solve Mwpmp, we employ the well-known GREEDYMATCHING algorithm [22], that selects the heaviest edge $e \in E$, removes e and edges incident to e, and reiterates until no edges are left. It is well-known that GREEDYMATCHING is a $\frac{1}{2}$-approximation for Mwmp [23], [22]. We just need to show that this performance guarantee holds for the Mwmp on $\mathcal{G}^{\text{LSAP}}$.

Lemma 4. GREEDYMATCHING *is a $\frac{1}{2}$-approximation for the Mwmp on $\mathcal{G}^{\text{LSAP}}$.*

Proof: (sketch) Because $\mathcal{G}^{\text{LSAP}}$ is complete and counts an even number of vertices ($2|\mathcal{T}^i|$), it is easy to see that GREEDYMATCHING returns a set of edges that cover all vertices. Thus, GREEDYMATCHING returns a perfect matching. This completes the proof. ■

Performance guarantee of HTA-GRE*:* We show that HTA-GRE is a $\frac{1}{8}$-approximation for the HTA problem.

Theorem 4. HTA-GRE *algorithm is a $\frac{1}{8}$-approximation for the* HTA *problem.*

Proof: (sketch): The proof structure is analogous to that of the approximation preserving reduction presented to prove Theorem 3. Once an instance of HTA is reduced to an instance of MAXQAP, the proof follows two parts. First, we show that the value of the optimal solution for HTA is less than 4 times the solution value for the auxiliary problem LSAP. Then, we show that the value of the solution returned by HTA-GRE is greater than $\frac{1}{2}$ times the value of the optimal solution for LSAP. Combining both propositions completes the proof. ■

Lemma 5. *The running time complexity of* HTA-GRE *is* $\mathcal{O}(|\mathcal{T}^i|^2 \log |\mathcal{T}^i|)$.

Proof: The time complexity of HTA-GRE is dominated by its two matching steps (Algorithm 2 - Lines 2 and 11). As observed by Arkin et al., the first matching step (Line 2) can be completed using a *greedy* matching. It is well-known that a greedy matching can be computed in $\mathcal{O}(|E| \log |V|)$ time on a graph $G = (V, E)$. The first matching runs on $|\mathcal{T}^i|$ vertices and $\frac{|\mathcal{T}^i|(|\mathcal{T}^i|-1)}{2}$ edges. The second matching (Line 11) runs on $2|\mathcal{T}^i|$ vertices and on $|\mathcal{T}^i|^2$ edges. The overall running time complexity is thus $\mathcal{O}(|\mathcal{T}^i|^2 \log |\mathcal{T}^i|)$. ■

V. EMPIRICAL VALIDATION

We run two kinds of experiments: (1) an offline simulation to stress the performance of our algorithms, and (2) an online deployment where we capture workers' motivation, perform task assignment accordingly, and measure end-to-end performance. The source code used to run our experiments is available at https://github.com/htacs/ata.

A. Summary of Results

Our first set of experiments is based on simulated workers and evaluates scalability and expected motivation, i.e., value of objective function. We find that HTA-GRE performs better than HTA-APP thanks to its greedy strategy that depends mainly on the number of tasks, *not* on the number of workers. The Hungarian algorithm on which HTA-APP relies is, on the other hand, more expensive. We find that HTA-GRE has an acceptable response time and could therefore be executed in the background while workers complete tasks, to prepare the next round of assignments. We also find that the greedy strategy of HTA-GRE does not hurt the value for the objective function when compared to HTA-APP. Therefore, HTA-GRE should be the algorithm of choice for the holistic task assignment problem as it runs faster while producing an assignment comparable to the one produced by HTA-APP.

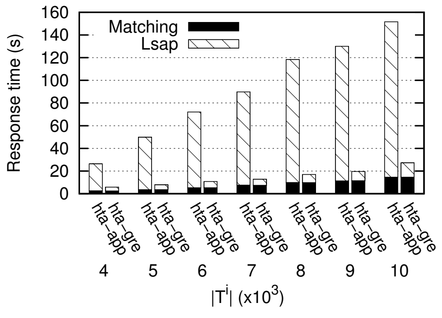
Our second experiments rely on effectively deploying tasks with real workers. The goal is to validate relevance and diversity as motivation factors and check the utility of adaptivity. We compare HTA-GRE with two non-adaptive strategies: one based on diversity only and another on relevance only. We show that optimizing diversity only results in the highest crowdwork quality. Relevance only, on the other hand, is consistently outperformed along quality, throughput and retention. *We also find that* HTA-GRE *offers the best compromise between those dimensions, i.e., crowdwork quality, task throughput and worker retention.*

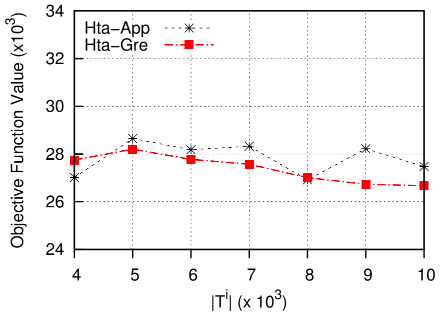
B. Offline Simulation

We focus on measuring scalability in terms of how fast tasks are assigned. Ideally, a task assignment algorithm should be quick and should also maximize workers' motivation. Therefore, we also evaluate the value of the objective function. We compare the performance of HTA-APP and HTA-GRE.

Implementation: We implement HTA-APP and HTA-GRE in Java and run them on Oracle's 1.8.0_91 JVM on a Debian 8.7 server with two Intel Xeon E5-2650@2.60 GHz and 128GB of RAM. For all sorting sub-routines, specifically when a *greedy* matching has to be computed, we use `Arrays.sort`, an improved implementation of the Merge Sort algorithm (aka "TimSort", runs in $\mathcal{O}(n \log n)$). To solve LSAP in HTA-APP, we adapt the C implementation of the Hungarian Algorithm of Carpaneto et al. [24] which runs in $\mathcal{O}(n^3)$ (available online [17]). We report the average of ten runs each time.

Data Sets: We use real tasks and synthetic workers. First, we crawled $152,221$ *task groups* from Amazon Mechanical Turk (AMT). Each task group contains id, title, reward, description, requester name, and keywords describing the metadata of all tasks in that group. As we vary the number of task groups and the number of tasks per group, we have #task groups $\times$ #tasks per task group $= |\mathcal{T}^i|$ tasks as input to


(a) X-axis: Number of tasks
$|\mathcal{W}^i| = 200$


(b) X-axis: Number of tasks
$|\mathcal{W}^i| = 200$

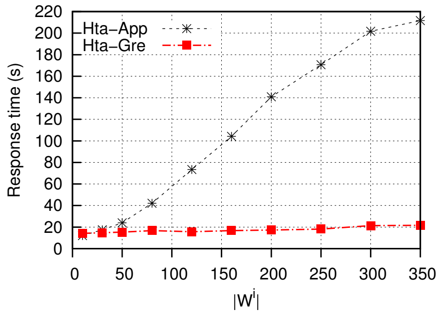

(c) X-axis: Number of workers
$|\mathcal{T}^i| = 8000$

Fig. 2. Scalability w.r.t. the number of tasks and workers (200 task groups, $X_{max} = 20$)

each experiment. Our workers are synthetically generated. For each worker w, we use a pseudo-random uniform generator to choose five keywords. Our goal is to evaluate the efficiency of our algorithms assuming that we have already captured workers' motivation. Therefore, we also need to simulate a previous iteration where workers have completed tasks: for each worker, we pick a random α_w^i and β_w^i in $[0, 1]$.

Number of Tasks: We vary the number of tasks $|\mathcal{T}^i|$ from $4,000$ to $10,000$, with 200 tasks per task group. Figure 2a shows the detailed response times of HTA-APP and HTA-GRE. As expected, HTA-APP is outperformed by HTA-GRE as its response time grows faster with the number of tasks. We can observe that the difference comes from the second phase of the algorithm that solves the auxiliary problem LSAP. For this phase, the response time of HTA-APP is $\mathcal{O}(|\mathcal{T}^i|^3)$ while HTA-GRE is $\mathcal{O}(|\mathcal{T}^i|^2 \log |\mathcal{T}^i|)$. The results in Figure 2b show that both HTA-APP and HTA-GRE report very similar values for the objective function, confirming the benefit of HTA-GRE.

Number of Workers: Figure 2c shows the response time as a function of the number of workers. Here again, HTA-GRE outperforms HTA-APP. The difference comes from the Hungarian Algorithm in HTA-APP whose response time increases with the number of workers. That is due to the number of *0-weight* edges used in its initialization phase. The more 0-weight edges, the higher the chance to find an initial solution that covers all vertices. This allows early termination and avoids running the expensive sub-routine `augment` to find an augmenting path [17] (which runs in $\mathcal{O}(|\mathcal{T}^i|^2)$ time). Even if the initial solution does not cover all vertices, more 0-weight edges also increase the chances of finding a long alternating path, favoring early termination. For instance, the number of calls to the `augment` sub-routine was 66 times higher when $|\mathcal{W}^i| = 350$ than when $|\mathcal{W}^i| = 30$. This is because many edges have the same weight when there are fewer workers (i.e. there are many $k, l \in 1, \ldots, |\mathcal{T}^i|$ where $f_{k,l} = 0$), which in turn incurs 0-values in the dual problem. The sub-routine GREEDYMATCHING in HTA-GRE was less affected by the number of workers since the response time of the sorting phase

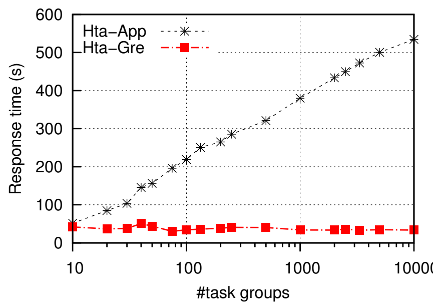

$|\mathcal{T}^i| = 10^3$, $|\mathcal{W}^i| = 300$, $X_{max} = 20$

Fig. 3. Effect of task diversity

in GREEDYMATCHING increases slowly with the number of workers. We also observed similar results for the value of the objective function.

Diversity of Tasks: We vary the number of task groups from 10 to $10,000$ with a fixed number of tasks $|\mathcal{T}^i| = 10,000$ (e.g. when there are $10,000$ task groups there is one task per task group). Here too, we see that HTA-GRE outperforms HTA-APP mainly because of the Hungarian Algorithm in HTA-APP (Figure 3). The more diverse the tasks, the more diverse the $f_{k,l}$ values in the LSAP problem, and the smaller the chance to build 0-weight edges in the dual problem. Therefore, the higher the number of diverse tasks, the less likely the Hungarian Algorithm will terminate early. Our algorithm HTA-GRE is oblivious to the diversity of tasks. Indeed, the running time of the sort phase in the GREEDYMATCHING sub-routine does not depend on the diversity of $f_{k,l}$ values.

C. Online deployment

The purpose of online deployment is to verify the behavior of HTA-GRE in practice and assess the utility of diversity to model motivation, and the need for adaptivity with real workers. We hence choose not to deploy HTA-APP. Each worker enters a work session during which several iterations occur. We measure 3 performance indicators: *outcome quality* that reflects the quality of a worker's contribution to a task

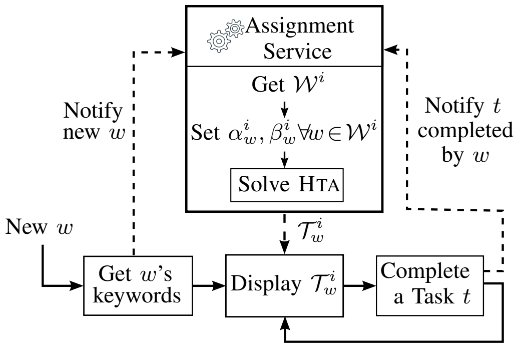


Fig. 4. Crowdsourcing workflow

against a ground truth, *task throughput* that measures the number of completed tasks per session and per unit of time, and *worker retention* that characterizes how long workers stay motivated during a work session.

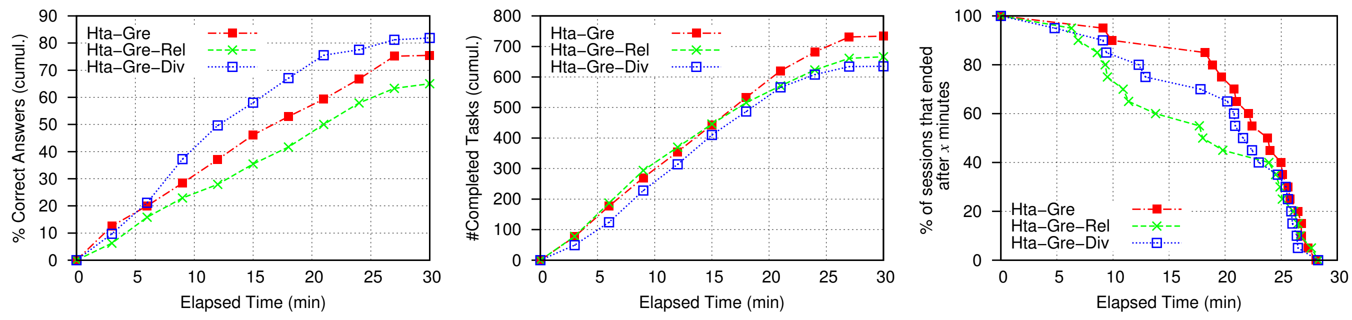
Set-up: We compare HTA-GRE with 2 other assignment strategies: (i) HTA-GRE-DIV, that uses HTA-GRE with $\alpha_w^i = 1, \beta_w^i = 0 \, \forall w \in \mathcal{W}^i$ hence optimizing diversity only, and (ii) HTA-GRE-REL, that uses HTA-GRE and sets $\alpha_w^i = 0, \beta_w^i = 1 \, \forall w \in \mathcal{W}^i$ hence optimizing relevance only. HTA-GRE is the only adaptive algorithm: it recomputes α_w^i and β_w^i for each worker in $\mathcal{W}^i$. For the first iteration of HTA-GRE, we assign X_{max} random tasks to address the cold start.

Workflow: To achieve adaptivity, we developed a new crowdsourcing platform where workers are hired from an external source, AMT in our case. Figure 4 illustrates a work session for a given worker w. First, we ask workers to choose at least 6 keywords and build their keyword vectors. Then, we display the first set of tasks $\mathcal{T}_w^i$ assigned to w. At each iteration, each worker w is shown a new set of tasks. We designed an assignment service that monitors all workers at once. When a new worker arrives, we notify the assignment service that a new set of tasks $\mathcal{T}_w^i$ has to be assigned to w. Then, each time w completes a task, we notify the assignment service, which in turn decides if a new assignment iteration must occur. It does so by monitoring the number of tasks completed overall and the number of tasks completed by each worker. Our rationale is: (i) to have a stable system, that does not re-assign tasks to workers too frequently, (ii) to get sufficient input to accurately estimate α_w^i and β_w^i for each worker and (iii) to define the set of available workers $\mathcal{W}^i$ at iteration i. We set $X_{max} = 15$ and display a total of 20 tasks per worker with an additional 5 random tasks to avoid falling into a silo by assigning only highly relevant tasks to each worker.

Tasks and Workers: We used a set of $158,018$ micro-tasks released by Crowdflower. It includes 22 different kinds of tasks, featuring for instance tweet classification, searching information on the web, transcription of images, sentiment analysis, entity resolution, and extracting information from news. Each kind of task is assigned a set of keywords that best describe its content and a reward, ranging from $\$0.01$ to $\$0.12$. Our tasks are *micro-tasks* that take less than a minute to complete. We published 160 HITs on AMT to recruit workers. Workers were then redirected to our platform to complete tasks in a HIT. Each HIT contains several tasks and defines one work session. Once done, workers receive a confirmation code to be submitted to AMT for payment. The HIT reward is set to $\$0.1$. Workers also get paid for every task completed. To qualify for our experiment, we required workers to have previously completed at least 100 HITs that were approved, and to have an approval rate above 80%. We also required HITs to be completed within 30 minutes. Then, we filtered results. First, we filtered 12 HITs where workers did not observe the alloted time: some stayed several hours. Second, we filtered 53 HITs where workers didn't complete at least one iteration. In 7 of this HITs, workers left after one task. The rationale is to (i) avoid comparing work sessions where workers are only assigned tasks using the cold-start random assignment (for HTA-GRE) and (ii) get significant results. We obtain 95 work sessions (including 20 for the HTA-GRE-DIV strategy). In order to make our strategies comparable, we selected the 20 work sessions with the highest number of completed tasks in each strategy. Overall, 58 different workers completed $2,715$ tasks in 80 work sessions for all 4 strategies.

Quality: We measure the quality of individual contributions from the crowd. $4,473$ questions were asked for the $2,715$ tasks that were completed (a task may have several questions). For a sample of $1,137$ questions, we used the ground truth provided by Crowdflower. Figure 5a presents the cumulative percentage of questions that were correctly completed for each strategy. We observe that workers performed better with HTA-GRE-DIV (81.9% of correct answers) than with other strategies (the significance level is 0.06 using two-proportions Z-test). HTA-GRE was slightly below (75.5%) and outperforms HTA-GRE-REL (significance level: 0.01) that has the worst ratio (65%). In fact, with HTA-GRE, the rate at which correct answers are provided is constant, while it starts to drop for HTA-GRE-REL after 21 minutes. This result validates our choice of using diversity in the expression of workers' motivation. It raises however the utility of adaptivity. *If workers perform best along the quality dimension with* HTA-GRE-DIV*, what do we need* HTA-GRE *for?*

Task throughput: Figure 5b presents the total number of completed tasks for each strategy. Overall, HTA-GRE outperforms all others: 734 tasks were completed with HTA-GRE, 666 with HTA-GRE-REL and 636 with HTA-GRE-DIV. When comparing HTA-GRE to HTA-GRE-DIV, the significance level is 0.05 using Mann-Whitney U test on the number of completed tasks per session. Also, the number of completed tasks for HTA-GRE grows steadily. This shows the superiority of HTA-GRE as it maintains workers' efficiency throughout their work session. At the very end, all curves decrease since workers need to submit their HIT before the allotted time expires (30 minutes). We observed that workers who were



(a) Crowdwork quality (b) Task throughput (c) Worker retention

Fig. 5. Results of online experiments

assigned tasks using HTA-GRE completed 36.7 tasks per work session, with an average session length of 22.3 minutes and an average task reward of \$0.064. *So far, in addition to showing that* HTA-GRE *is clearly superior along the task throughput dimension, we show that while* HTA-GRE-DIV *provided the best outcome quality, it has the worst task throughput. We conjecture that too much diversity results in overhead in choosing tasks. Similarly, providing relevant tasks only may induce boredom. A balance needs to be found with adaptivity.*

Worker retention: Figure 5c shows worker retention as the percentage of work sessions (vertical axis) that ended after x minutes (horizontal axis). This measure is akin to a *survival rate*, showing if workers are motivated to stay longer in a session. We observe that HTA-GRE outperforms HTA-GRE-REL and HTA-GRE-DIV (significance level is 0.1 using a Mann-Whitney U test on sessions durations). For instance, 85% of workers stayed over 18.2 minutes with HTA-GRE. We can therefore answer our initial question: HTA-GRE *offers the best compromise between crowdwork quality and overall performance in terms of task throughput and worker retention.*

VI. Related Work

Worker Motivation: Worker motivation was first studied in physical workplaces [7]. Recent studies [8] investigated 13 motivation factors for workers and found that workers were interested in *skill variety* or *task autonomy* as much as *task reward*. A range of studies point out the importance of suitably motivating workers in crowdsourcing [25], [26]. Kittur et al. [26] underlined the interest of designing frameworks that include incentive schemes other than financial ones. In particular, they notice that a system should "*achieve both effective task completion and worker satisfaction*".

Some efforts were driven towards experimenting with motivation factors in *task completion* [10], [12], [27]. In a recent study, [12] proposed *diversions* in the workflow such that workers were presented with some entertainment that improves worker retention. Chandler and Kapelner [10] empirically showed that workers perceived "meaningfulness" of a task improved throughput without degrading quality. Shaw et al. [27] assessed 14 incentives schemes and found that incentives based on worker-to-worker comparisons yield better crowdwork quality. Hata et al. [28] studied how work quality changes over extended periods of time. None of the above studies leverages motivation to optimize task assignment.

Adaptive Task Assignment: Adaptive task assignment has been studied with a particular focus on maximizing the quality of crowdwork [1], [3], [4], [29]. For instance, Fan et al. [1] leverage similarity between tasks and workers' past contributions to maximize crowdwork accuracy. Ho et al. [3] study an online setting, where workers' skills are learned and used in task assignment. None of these studies includes motivation factors and or task diversity in their model. More recently, it has been shown that leveraging motivation in individual task assignment [30] improves crowdsourcing performance, particularly, crowdwork quality. We differ from that work by studying task assignment holistically and we use a formulation for motivation that captures diversity and relevance. In [31], [32], the authors study the problem in the context of spatial crowdsourcing with constraints on time, budget, and distance. The strategy they propose leverages space partitioning techniques as well as arrival orders of tasks or workers. Our problem does not assume a spatial setting - hence the techniques proposed in these papers do have any easy adaptation to solve our problem.

Diversity Formulations and Algorithms: Diversity is a widely studied subject that finds its roots in Web search with a goal similar to ours. For example, it was used in text retrieval and summarization to balance document relevance and novelty [33]. Most approaches fall into two cases: *content-based* (e.g. [33]) and *intent-based* [34]. Gollapudy et al. [13] adopt an axiomatic approach to diversity to address user intent. They show that no diversity function can satisfy all axioms together. Our formulation is content-based as it relies on a distance function between tasks. Other content-based functions, such as ones based on taxonomies, are possible [35]. In our work, it is essential to use a metric to obtain our approximation results.

In the database context, Chen and Li [36] propose to post-process structured query results to enforce diversity. Similarly, in recommendations [37], [38], intermediate results are post-processed, using pairwise item similarity, to generate accurate and diverse recommendations. In [39], a hierarchical notion of diversity in databases is introduced, and efficient top-k processing algorithms are proposed. In [14], an algorithm with provable approximation guarantees is proposed to find relevant and diverse news articles.

VII. Conclusion and Future Work

We formulated and solved motivation-aware task assignment for a set of workers and tasks. We defined motivation as a balance between task relevance and diversity. Our approach is adaptive as it relies on observing workers in task completion, capturing their motivation and using it to determine the next tasks to assign to them. At each iteration, we addressed HTA, the holistic task assignment problem that finds an assignment of available tasks to available workers such that the total expected worker motivation is maximized. We showed that HTA is NP-Hard and further proved that it is Max-SNP-Hard. We developed HTA-APP and HTA-GRE, two algorithms that run in polynomial time and have $1/4$ and $1/8$ approximation factors, respectively. Our offline experiments examined response time and the value of relevance and diversity for both algorithms. We showed that HTA-GRE is faster than HTA-APP, without compromising motivation. We also deployed online experiments on our home-grown platform and demonstrated the necessity to account for diversity and relevance in formulating workers' motivation, and the high quality of individual worker contributions with adaptive task assignment. Our immediate plan is to extend this work to collaborative tasks where motivation factors such as social signaling matter. Task assignment would have to account for the presence of other workers in forming the most motivated team to complete a task. This makes task assignment challenging as it needs to be streamed and will depend on the availability of workers with complementary skills.

References

[1] J. Fan *et al.*, "icrowd: An adaptive crowdsourcing framework," in *SIGMOD*, 2015.
[2] Y. Gao *et al.*, "Finish them!: Pricing algorithms for human computation," *PVLDB*, 2014.
[3] C. Ho *et al.*, "Adaptive task assignment for crowdsourced classification," in *ICML*, 2013.
[4] C. Ho and J. W. Vaughan, "Online task assignment in crowdsourcing markets," in *AAAI*, 2012.
[5] H. Rahman *et al.*, "Task assignment optimization in collaborative crowdsourcing," in *IEEE ICDM*, 2015, pp. 949–954.
[6] S. B. Roy *et al.*, "Task assignment optimization in knowledge-intensive crowdsourcing," *VLDB J.*, 2015.
[7] J. Hackman and G. R. Oldham, "Motivation through the design of work: Test of a theory," *Organizational Behavior and Human Performance*, 1976.
[8] N. Kaufmann, T. Schulze, and D. Veit, "More than fun and money. worker motivation in crowdsourcing - A study on mechanical turk," in *AMCIS*, 2011.
[9] J. Rogstadius *et al.*, "An assessment of intrinsic and extrinsic motivation on task performance in crowdsourcing markets," in *ICWSM*, 2011.
[10] D. Chandler and A. Kapelner, "Breaking monotony with meaning: Motivation in crowdsourcing markets," *CoRR*, vol. abs/1210.0962, 2012.
[11] J. J. Horton and L. B. Chilton, "The labor economics of paid crowdsourcing," in *ACM EC*, 2010, pp. 209–218.
[12] P. Dai *et al.*, "And now for something completely different: Improving crowdsourcing workflows with micro-diversions," in *ACM CSCW*, 2015.
[13] S. Gollapudi and A. Sharma, "An axiomatic approach for result diversification," in *WWW*, 2009, pp. 381–390.
[14] S. Abbar *et al.*, "Real-time recommendation of diverse related articles," in *WWW*, 2013, pp. 1–12.
[15] T. A. Feo and M. Khellaf, "A class of bounded approximation algorithms for graph partitioning," *Networks*, pp. 181–195, 1990.
[16] E. M. Arkin *et al.*, "Approximating the maximum quadratic assignment problem," *Inf. Process. Lett.*, pp. 13–16, 2001.
[17] R. E. Burkard *et al.*, *Assignment Problems*. SIAM, 2009.
[18] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.
[19] A. Besicovitch, "On a general metric property of summable functions," *Journal of the London Mathematical Society*, 1926.
[20] V. Nagarajan and M. Sviridenko, "On the maximum quadratic assignment problem," in *SODA*.
[21] G. Ausiello *et al.*, "Approximation preserving reductions," in *Complexity and Approximation*. Springer, 1999.
[22] R. Duan and S. Pettie, "Linear-time approximation for maximum weight matching," *J. ACM*, 2014.
[23] D. E. Drake and S. Hougardy, "A simple approximation algorithm for the weighted matching problem," *Inf. Process. Lett.*, pp. 211–213, 2003.
[24] G. Carpaneto *et al.*, "Algorithms and codes for the assignment problem," *Annals of operations research*, pp. 191–223, 1988.
[25] B. B. Bederson and A. J. Quinn, "Web workers unite! addressing challenges of online laborers," in *CHI*, 2011, pp. 97–106.
[26] A. Kittur *et al.*, "The future of crowd work," in *CSCW*, 2013.
[27] A. D. Shaw *et al.*, "Designing incentives for inexpert human raters," in *CSCW*, 2011.
[28] K. Hata *et al.*, "A glimpse far into the future: Understanding long-term crowd worker quality," in *CSCW*, 2017.
[29] Y. Zheng *et al.*, "QASCA: A quality-aware task assignment system for crowdsourcing applications," in *SIGMOD*, 2015.
[30] J. Pilourdault, S. Amer-Yahia, D. Lee, and S. B. Roy, "Motivation-aware task assignment in crowdsourcing," in *EDBT*, 2017, pp. 246–257.
[31] P. Cheng, X. Lian, L. Chen, J. Han, and J. Zhao, "Task assignment on multi-skill oriented spatial crowdsourcing," *IEEE TKDE*, vol. 28, no. 8, pp. 2201–2215, 2016.
[32] Y. Tong, J. She, B. Ding, L. Wang, and L. Chen, "Online mobile micro-task allocation in spatial crowdsourcing," in *ICDE*, 2016, pp. 49–60.
[33] J. G. Carbonell and J. Goldstein, "The use of MMR, diversity-based reranking for reordering documents and producing summaries," in *Research and Development in Information Retrieval*, 1998.
[34] O. Chapelle *et al.*, "Intent-based diversification of web search results: metrics and algorithms," *Information Retrieval*, pp. 572–592, 2011.
[35] A. Anagnostopoulos *et al.*, "Sampling search-engine results," *WWW*, 2006.
[36] Z. Chen and T. Li, "Addressing diverse user preferences in sql-query-result navigation," in *SIGMOD*, 2007, pp. 641–652.
[37] C. Ziegler *et al.*, "Improving recommendation lists through topic diversification," in *WWW*, 2005, pp. 22–32.
[38] K. El-Arini, G. Veda, D. Shahaf, and C. Guestrin, "Turning down the noise in the blogosphere," in *SIGKDD*, 2009, pp. 289–298.
[39] E. Vee *et al.*, "Efficient computation of diverse query results," in *ICDE*, 2008, pp. 228–236.

Appendix

A. *Proof of Theorem 4*

We adapt the proof of Arkin et al. [16]. First, we show that the value of the optimal solution for HTA is less than 4 times the solution value for the auxiliary problem LSAP.

Let M_B be a maximum matching in B. Let $\overline{M_B} = \{\{t_k, t_l\} \notin M_B,\ \exists t_{k'} s.t. \{t_k, t_{k'}\} \in M_B, \exists t_{l'} s.t. \{t_l, t_{l'}\} \in$

$M_B\}$. Since M_B is a maximum matching, we have

$$\forall t_k, t_l \in \overline{M_B}, d(t_k, t_l) \leq d(t_k, t_{k'}) + d(t_l, t_{l'}) \tag{9}$$

Let t_m be the task that is not incident to any edge in M_B (there is at most such one, when $|\mathcal{T}^i|$ is odd). Let $E(t_m)$ the set of edges incident to t_m. Since M_B is a maximum matching:

$$\forall \{t_n, t_{n'}\} \in M_B, d(t_m, t_n) \leq d(t_n, t_{n'}) \wedge d(t_m, t_{n'}) \leq d(t_n, t_{n'}) \tag{10}$$

Both Equations 9 and 10 also hold if M_B is a *greedy* matching [16]. Let $\mathcal{Y} = \{1, \ldots, |\mathcal{T}^i|\}$ and π^* be the permutation of $\mathcal{Y}$ associated to the optimal solution $\textsf{hta}^\star$ of HTA. We have:

$$\textsf{hta}^\star = \sum_{\substack{k,l \in \mathcal{Y} \\ k \neq l}} a_{\pi^*(k),\pi^*(l)} b_{k,l} + \sum_{k \in \mathcal{Y}} c_{k,\pi^*(k)} \tag{11}$$

We decompose the first member of the sum:

$$\sum_{\substack{k,l \in \mathcal{Y} \\ k \neq l}} a_{\pi^*(k),\pi^*(l)} d(t_k, t_l) = \sum_{\substack{k,l \in \mathcal{Y}, \\ \{t_k,t_l\} \in M_B}} a_{\pi^*(k),\pi^*(l)} d(t_k, t_l) + \\ + \sum_{\substack{k,l \in \mathcal{Y}, \\ \{t_k,t_l\} \in \overline{M_B}}} a_{\pi^*(k),\pi^*(l)} d(t_k, t_l) + \sum_{\substack{k,l \in \mathcal{Y}, \\ \{t_k,t_l\} \in E(t_m)}} a_{\pi^*(k),\pi^*(l)} d(t_k, t_l) \tag{12}$$

Now,

$$\sum_{\substack{k,l \in \mathcal{Y}, \\ \{t_k,t_l\} \in \overline{M_B}}} a_{\pi^*(k),\pi^*(l)} d(t_k, t_l) \\ \leq \sum_{\substack{k,l \in \mathcal{Y}, \\ \{t_k,t_l\} \in \overline{M_B}}} a_{\pi^*(k),\pi^*(l)} (d(t_k, t_{k'}) + d(t_l, t_{l'})) \tag{13}$$

where $\{t_k, t'_k\} \in M_B$ and $\{t_l, t'_l\} \in M_B$ (using Equation 9). Additionally,

$$\sum_{\substack{k,l \in \mathcal{Y}, \\ \{t_k,t_l\} \in E(t_m)}} a_{\pi^*(k),\pi^*(l)} d(t_k, t_l) \leq \sum_{\substack{k,l \in \mathcal{Y}, \\ \{t_k,t_l\} \in E(t_m)}} a_{\pi^*(k),\pi^*(l)} d(t_n, t'_n) \tag{14}$$

where $\{t_n, t'_n\}$ is the edge in M_B incident to $\{t_k, t_l\} \in \overline{M_B}$ (using Equation 10). Combining Equations 13 and 14:

$$\sum_{\substack{k,l \in \mathcal{Y} \\ k \neq l}} a_{\pi^*(k),\pi^*(l)} d(t_k, t_l) \leq \sum_{\substack{k,l \in \mathcal{Y}, \\ \{t_k,t_l\} \in M_B}} d(t_k, t_l)(deg^A_{\pi^*(k)} + deg^A_{\pi^*(l)}) \tag{15}$$

Using Equation 15 in Equation 11:

$$\textsf{hta}^\star \leq \sum_{\substack{k,l \in \mathcal{Y}, \\ \{t_k,t_l\} \in M_B}} d(t_k, t_l)(deg^A_{\pi^*(k)} + deg^A_{\pi^*(l)}) + 2 \sum_{k \in \mathcal{Y}} c_{k,\pi^*(k)} \\ \leq 2 \sum_{k \in \mathcal{Y}} f_{k,\pi^*(k)} \tag{16}$$

Let π'^* the optimal solution of the LSAP instance and π' the solution of GREEDYMATCHING on this instance. Let $\textsf{lsap-sol}$ the value of the solution π' for LSAP. We have:

$$\textsf{lsap-sol} = \sum_{k \in \mathcal{Y}} f_{k,\pi'(k)} \geq \frac{1}{2} \sum_{k \in \mathcal{Y}} f_{k,\pi'^*(k)} \tag{17}$$

since GREEDYMATCHING is a $\frac{1}{2}$-approximation for this LSAP instance. Therefore, we have

$$\textsf{hta}^\star \leq 2 \sum_{k \in \mathcal{Y}} f_{k,\pi^*(k)} \leq 4 \sum_{k \in \mathcal{Y}} f_{k,\pi'(k)} \leq 4 * \textsf{lsap-sol} \tag{18}$$

We now prove the second part of the proof. Let $\textsf{hta-gre-sol}$ be the value of the solution returned by HTA-GRE for HTA. Let π the permutation associated to this solution. We have:

$$\textsf{hta-gre-sol} = \sum_{\substack{k,l \in \mathcal{Y}, \\ \{t_k,t_l\} \in M_B}} a_{\pi(k),\pi(l)} d(t_k, t_l) + \sum_{\substack{k,l \in \mathcal{Y}, \\ \{t_k,t_l\} \in \overline{M_B}}} a_{\pi(k),\pi(l)} d(t_k, t_l) \\ + \sum_{\substack{k,l \in \mathcal{Y}, \\ \{t_k,t_l\} \in E(t_m)}} a_{\pi(k),\pi(l)} d(t_k, t_l) + \sum_{k \in \mathcal{Y}} c_{k,\pi(k)} \tag{19}$$

In Lines 12-16, HTA-GRE sets $(\pi(k), \pi(l))$ to $(\pi'(k), \pi'(l))$ or to $(\pi'(l), \pi'(k))$ for each $\{t_k, t_l\} \in M_B$. Each case occurs with probability $\frac{1}{2}$. In Equation 19, $a_{\pi(k),\pi(l)} = a_{\pi'(k),\pi'(l)}$ (A is symmetric and $t_k, t_l \in M_B$). Therefore, the expected contribution of pair $(k,l) \in \mathcal{Y}^2, \{t_k, t_l\} \in M_B$ is $a_{\pi'(k),\pi'(l)} * d(t_k, t_l)$. In Equation 19, each $d(t_k, t_l)$ is multiplied by $a_{\pi'(k),\pi'(l)}$ or $a_{\pi'(k'),\pi'(l)}$ or $a_{\pi'(k),\pi'(l')}$ or $a_{\pi'(k'),\pi'(l')}$ where $\{t_k, t_l\} \in \overline{M_B}, \{t_k, t_{k'}\} \in M_B, \{t_l, t_{l'}\} \in M_B$. Conversely, $a_{\pi'(k),\pi'(l)}$ is multiplied by $d(t_k, t_l)$ or $d(t_{k'}, t_l)$ or $d(t_k, t_{l'})$ or $d(t_{k'}, t_{l'})$, each case with probability $\frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$. Thus, the expected contribution of pair $(k,l) \in \mathcal{Y}^2, \{t_k, t_l\} \in \overline{M_B}$:

$$\frac{1}{4} a_{\pi'(k),\pi'(l)} (d(t_k, t_l) + d(t_{k'}, t_l) + d(t_k, t_{l'}) + d(t_{k'}, t_{l'})) \\ \geq \frac{1}{4} a_{\pi'(k),\pi'(l)} \times 2 \max\{d(t_k, t_{k'}), d(t_l, t_{l'})\} \\ \geq \frac{1}{4} a_{\pi'(k),\pi'(l)} (d(t_k, t_{k'}) + d(t_l, t_{l'})) \tag{20}$$

In Equation 19, each $a_{\pi'(k),\pi'(l)}$ where $k = m$ or $l = m$ (suppose $k = m$) is multiplied by $d(t_m, t_l)$ or $d(t_m, t'_l)$ where $\{t_l, t_{l'}\} \in M_B$, each with probability $\frac{1}{2}$. Therefore, the expected contribution of pair $(k,l) \in \mathcal{Y}^2, \{t_k, t_l\} \in E(t_m)$ is:

$$\frac{1}{2} a_{\pi'(k),\pi'(l)} (d(t_m, t_l) + d(t_m, t'_l)) \geq \frac{1}{2} a_{\pi'(k),\pi'(l)} * d(t_l, t_{l'}) \tag{21}$$

since $d()$ satisfies the triangular inequality. In 19, the expected contribution of $c_{k,\pi(k)}$ is at least $\frac{1}{2} c_{k,\pi'(k)}$ since $\pi(k) = \pi'(k)$ with probability $\frac{1}{2}$. We obtain:

$$\textsf{hta-gre-sol} \geq \frac{1}{4} \sum_{\substack{k,l \in \mathcal{Y}, \\ \{t_k,t_l\} \in M_B}} d(t_k, t_l)(deg^A_{\pi'(k)} + deg^A_{\pi'(l)}) + \frac{1}{2} \sum_{k \in \mathcal{Y}} c_{k,\pi'(k)} \\ \geq \frac{1}{2} \sum_{k \in \mathcal{Y}} f_{k,\pi'(k)} \tag{22}$$

Combining Equations 18 and 22, we prove:

$$\textsf{hta-gre-sol} \geq \frac{1}{2} \textsf{lsap-sol} \geq \frac{1}{8} \textsf{hta}^\star$$