2019 IEEE 35th International Conference on Data Engineering (ICDE)

SLADE: A Smart Large-Scale Task Decomposer in Crowdsourcing

Yongxin Tong [1], Lei Chen [2], Zimu Zhou [3], H. V. Jagadish [4], Lidan Shou [5], Weifeng Lv [1]
[1]BDBC, SKLSDE Lab and IRI, Beihang University, China
[2]Hong Kong University of Science and Technology, Hong Kong, [3]ETH Zurich, Switzerland
[4]University of Michigan, USA, [5]Zhejiang University, China
[1]{yxtong, lwf}@buaa.edu.cn, [2]leichen@cse.ust.hk, [3]zimu.zhou@tik.ee.ethz.ch [4]jag@umich.edu, [5]should@zju.edu.cn

***Abstract*—A crowdsourcing task in real-world applications often consists of thousands of atomic tasks. A common practice to distribute a large-scale crowdsourcing task is to pack atomic tasks into task bins and send to crowd workers in batches. It is challenging to decompose a large-scale crowdsourcing task into task bins to ensure reliability at a minimal total cost. In this paper, we propose the *Smart Large-scAle task DEcomposer (SLADE)* problem, which aims to decompose a large-scale crowdsourcing task to achieve the desired reliability at a minimal cost. We prove its NP-hardness and study two variants of the problem. For the homogeneous SLADE problem, we propose a greedy algorithm and an approximation framework using an optimal priority queue (OPQ) structure with provable approximation ratio. For the heterogeneous SLADE problem, we extend this framework and prove its approximation guarantee. Extensive experiments validate the effectiveness and efficiency of the solutions.**

I. Introduction

Crowdsourcing applications are increasingly popular in recent years. A real-world crowdsourcing task can contain thousands of *atomic tasks*, where an atomic task is a unit task that requires trivial cognitive load, *e.g.*, binary choices. A common practice to distribute a large-scale task to crowd workers is to pack a set of atomic tasks into a *task bin* and send to a crowd worker in a batch. Packing atomic tasks into task bins also reduces the average cost per atomic task

The size (*cardinality*) of the task bins is crucial for the execution plan of a large-scale crowdsourcing task in terms of *cost* and *reliability*. Decomposing a large-scale crowdsourcing task into task bins of a larger size results in a lower average cost of each atomic task in the task bins. Yet the overall reliability of a large batch of atomic tasks may decrease due to the increase of cognitive load. Hence these atomic tasks have to be distributed for more times to meet the reliability requirement of the entire task, which leads to an increased total cost. Previous works either set the fixed cardinality of a task bin [1] or adopt simple heuristics to determine a single cardinality for the entire large-scale crowdsourcing task.

In this paper, we propose to harness *a set of task bin cardinalities* rather than a single one to reduce the total cost in executing a large-scale crowdsourcing task while retaining the desired reliability. The key insight is that with the increase of the task bin cardinality, there is a mismatch between the drop of per atomic task reliability and that of per atomic task cost. On this basis, we propose the *Smart Large-scAle task DEcomposer (SLADE)* problem. It aims to decompose a crowdsourcing task into task bins of varied sizes, which achieves the reliability of each atomic task at a minimal total cost. The *SLADE* problem resembles a database query optimizer that finds an efficient execution plan given a logical expression to be evaluated. Our main contributions are: (1) We identify the two variants of the SLADE problem and prove their NP-hardness. (2) For the *homogeneous* SLADE problem, we propose a greedy heuristic and an optimal priority queue-based approximation algorithm with $\log n$-approximation ratio, where n is the number of all atomic tasks. For the *heterogeneous SLADE* problem, we extend the approximation framework and guarantees a slightly lower approximation ratio. (3) We extensively evaluate the effectiveness and efficiency of the algorithms on real datasets.

II. Problem Statement

A. Preliminaries

We focus on large-scale crowdsourcing tasks consisting of ***atomic tasks***. An atomic task, denoted by a_i, is defined as a binary choice problem. Binary atomic tasks dominate the types of task operations adopted in the marketplace. A ***large-scale crowdsourcing task*** T as defined as n independent atomic tasks, *i.e.*, $T = \{a_1, a_2, \cdots, a_n\}$ ($n = |T|$).

An ***l-cardinality task bin*** is a triple, denoted as $b_l = \langle l, r_l, c_l \rangle$, where (1) the cardinality l is the maximum number of atomic tasks that can be included in the task bin; (2) r_l is the confidence, which indicates the average probability that crowd workers can correctly complete each atomic task in this task bin; (3) c_l is the incentive cost given to the crowds who complete all the atomic tasks in this task bin.

We define the *reliability of an atomic task* as the probability of no false negatives, since many real-world crowdsourcing applications require low false negative ratios. Then we can link the reliability of an atomic task to the confidences of the task bins where the atomic task is assigned. Given an atomic task a_i and the set of assigned task bins $\mathfrak{B}(a_i)$, the reliability, denoted by $Rel(a_i, \mathfrak{B}(a_i))$, of a_i in $\mathfrak{B}(a_i)$ is defined as $Rel(a_i, \mathfrak{B}(a_i)) = 1 - \prod_{\beta \in \mathfrak{B}(a_i)} (1 - r_{|\beta|})$ where $|\beta|$ is the cardinality of the task bin β, and $r_{|\beta|}$ is its confidence.

B. Problem Formulation

Definition 1 (SLADE Problem)**.** *Given a large-scale crowdsourcing task T consisting of n atomic tasks $\{a_1, \ldots, a_n\}$, the corresponding reliability thresholds $\{t_1, \ldots, t_n\}$ for each*

2375-026X/19/$31.00 ©2019 IEEE
DOI 10.1109/ICDE.2019.00261

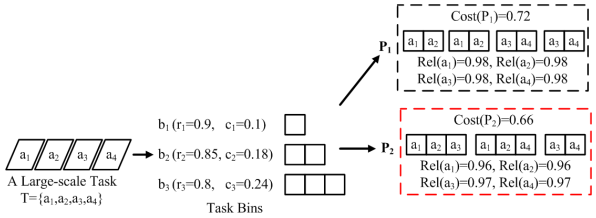

Fig. 1: Illustration of the SLADE Problem

atomic task, and a set of task bins $B = \{b_1, \ldots, b_m\}$*, the SLADE Problem is to find a planning* $DP_T = \{\tau_i, b_i\}_{i=1}^{m}$*, which means task bin* b_i *is used for* τ_i *times, to minimize the total cost,* $\sum_{i=1}^{m} \tau_i c_i$*, such that* $Rel(a_i, \mathfrak{B}(a_i)) \geq t_i$*,* $\forall a_i \in T$*.*

If the reliability threshold t_i of the atomic tasks are the same, the problem is called the ***homogeneous SLADE problem***. Otherwise, the problem is called the ***heterogeneous SLADE problem***. Each atomic task can be assigned to multiple l-cardinality task bins in a decomposition plan, *i.e.*, each atomic task can be dispatched to and processed by multiple crowd workers to improve the reliability of each atomic task.

Example 1. *(Homogeneous SLADE Problem) Fig. 1 shows a crowdsourcing task* $T = \{a_1, a_2, a_3, a_4\}$*, which contains four atomic tasks and a set of task bins* $B = \{b_1, b_2, b_3\}$*. The reliability thresholds of each atomic task* t_i *is 0.95 (*$1 \leq i \leq 4$*). A decomposition plan,* P_1*, is to adopt four 2-cardinality task bins:* $\{a_1, a_2\}$*,* $\{a_1, a_2\}$*,* $\{a_3, a_4\}$ *and* $\{a_3, a_4\}$*. In* P_1*, the reliability of* a_i *(*$1 \leq i \leq 4$*) is* $1 - (1 - 0.85) \times (1 - 0.85) = 0.98 > 0.95$*, with a total cost of* $0.18 \times 4 = 0.72$*. Another decomposition plan,* P_2*, is to use two 3-cardinality task bins and one 2-cardinality task bin:* $\{a_1, a_2, a_3\}$*,* $\{a_1, a_2, a_4\}$ *and* $\{a_3, a_4\}$*. The reliability of the atomic tasks also exceeds* 0.95*, but the cost is only* $0.24 \times 2 + 0.18 = 0.66$*. Fig. 1 illustrates the two decomposition plans. In fact,* P_2 *is the optimal decomposition plan for this example.*

III. Algorithm Framework

A. Solution to Homogeneous SLADE Problem

Greedy Algorithm. It considers the cost-confidence ratio of each task bin as its selection criterion. Specifically, the cost-confidence ratio for an l-cardinality task bin and its corresponding atomic tasks is defined as $ratio = \frac{c_l}{min\{l \times (-\ln(1-r_l)), \sum_{k=1}^{l} \theta_{i_k}\}}$. The algorithm always selects the task bin and the corresponding atomic tasks with the lowest ratio into the decomposition plan until the all the atomic tasks satisfy the reliability constraint.

OPQ-based Algorithm. It is an approximation algorithm based on a data structure called the optimal priority queue (OPQ). The algorithm consists of two steps.

Step 1. **Constructing OPQ.** Given a set of task bins $B = \{b_1, \cdots, b_m\}$ and a reliability threshold t, an optimal priority queue OPQ is a priority queue consisting of the combinations of task bins ($Comb$'s) and satisfies the following conditions: (1) the elements in OPQ is ranked in an descending order of their corresponding LCM values; (2) for any element OPQ_i (with $OPQ_i.LCM$ and $OPQ_i.UC$) in OPQ, there is no element OPQ_j (with $OPQ_j.LCM$ and $OPQ_j.UC$) such that $OPQ_i.LCM \geq OPQ_j.LCM$ and $OPQ_i.UC \geq OPQ_j.UC$; (3) all the combinations of task bins in OPQ satisfy the reliability threshold for each atomic task. We design a depth-first-search-based enumeration algorithm to construct the OPQ.

Step 2. **OPQ-Based Task Decomposition.** In this step, we repeatedly use the optimal combinations in the optimal priority queue to approximate the global optimal solution.

B. Solution to Heterogeneous SLADE Problem

We extend the solutions to the homogeneous SLADE problem into the heterogenous scenario.

Extended Greedy Algorithm. The greedy algorithm for the heterogeneous scenario is the same as that for the homogeneous scenario, except that the reliability thresholds of the atomic tasks are now different, which affects the threshold residual θ_{i_k} of the above cost-confidence ratio.

Extended OPQ-based Algorithm. We partition the whole set of atomic tasks into groups and run OPQ-based algorithm for each group. Specifically, we use quantiles of $2^{\alpha+i}$ to divide the range of the thresholds into different intervals.

IV. Experimental Study

Datasets. We conduct experiments on two real datasets gathered by running tasks on Amazon MTurk. The first dataset is gathered from the *jelly-beans-in-a-jar* experiments [2] and the second is from the *micro-expression identification* experiments [3]. We set the default maximum cardinality ($|B|$) to 20, and the number of atomic tasks to 10,000. In the homogenous scenarios, the reliability threshold t is set to 0.9 for all atomic tasks. In the heterogeneous scenarios, the default reliability thresholds are generated according to the Normal distribution with parameters μ and σ set to 0.9 and 0.03, respectively.

Compared Methods. We compare *Baseline*, *Greedy*, and *(Extended) OPQ-Based* algorithms for the homogenous scenarios and the heterogeneous scenarios.

Summary of Experimental Results. For the homogeneous scenario, *OPQ-Based* is both the most effective and efficient. *Baseline* is the least effective and *Greedy* is the least efficient. For the heterogeneous scenario, when the number of distinct reliability thresholds increases, the three algorithms spend more running time. When the number of lower reliability thresholds increases, the decomposition cost will decrease.

Acknowledgment

This work is partially supported by National Science Foundation of China (NSFC) under Grant No. 61822201, U1811463, and Didi Gaia Collaborative Research Funds for Young Scholars.

References

[1] M. J. Franklin, D. Kossmann, T. Kraska, S. Ramesh, and R. Xin, "CrowdDB: answering queries with crowdsourcing," in *SIGMOD*, 2011.
[2] Y. Tong, L. Chen, Z. Zhou, H. V. Jagadish, L. Shou, and W. Lv, "SLADE: A Smart Large-Scale Task Decomposer in Crowdsourcing," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 8, pp. 1588–1601, 2018.
[3] T. Pfister, X. Li, G. Zhao, and M. Pietikäinen, "Recognising spontaneous facial micro-expressions," in *ICCV*, 2011.