Task-Optimized Group Search for Social Internet of Things

Chih-Ya Shen*, Hong-Han Shuai†, Kuo-Feng Hsu‡, Ming-Syan Chen‡
*National Tsing Hua University, Hsinchu, Taiwan
†National Chiao Tung University, Hsinchu, Taiwan
‡National Taiwan University, Taipei, Taiwan
chihya@cs.nthu.edu.tw, hhshuai@nctu.edu.tw, {r03921038,mschen}@ntu.edu.tw

ABSTRACT

With the maturity and popularity of Internet of Things (IoT), the notion of Social Internet of Things (SIoT) has been proposed to support novel applications and networking services for the IoT in more effective and efficient ways. Although there are many works for SIoT, they focus on designing the architectures and protocols for SIoT under the specific schemes. How to efficiently utilize the collaboration capability of SIoT to complete complex tasks remains unexplored. Therefore, we propose a new query, namely *Task-Optimized Group Search (TOGS)*, to address this need. TOGS aims to extract the target SIoT group such that the target SIoT group will be able to easily communicate with each other while maximizing the accuracy of performing the given tasks. We propose two problem formulations, namely *Bounded Communication-loss TOSS (BC-TOSS)* and *Robustness Guaranteed TOSS (RG-TOSS)*, for different communication scenarios, and prove that they are both NP-Hard and inapproximable. We propose a polynomial-time algorithm with performance bound for BC-TOSS, and an efficient polynomial-time algorithm to obtain good solutions for RG-TOSS. The experimental results on real datasets indicate that our proposed algorithms outperform other baselines.

1. INTRODUCTION

With the maturity and popularity of Internet of Things (IoT), it has been widely recognized that the Internet of Things (IoT) is the next paradigm shift. The future Internet will embody a tremendous number of objects that provide valuable information and controllable actions. Moreover, with the capability of interactions among each other, objects can collaborate with other counterparts toward providing services to the end users, e.g., environmental monitoring, surveillance, smart home, health care, and product management.

Recently, since it has been shown that a large number of users tied in a social network can provide far more accurate answers to complex problems than a single user [18], a recent line of studies investigates the opportunities of integrating social networking concepts into solving complex problems. Several schemes have been proposed to exploit social networks for question answering via crowdsourcing [1], P2P routing [3], or web security [22]. Meanwhile, by incorporating the concept of social network into IoT, the idea of Social Internet of Things (SIoT) has been proposed to support novel applications and networking services for the IoT in more effective and efficient ways.

However, current research focuses on designing the architectures and protocols for SIoT under the specific schemes. For example, Kosmatos et al. integrated the RFID and smart object-based infrastructures towards building blocks of SIoT [8]. Nitti et al. proposed two trustworthiness management models to suggest strategies of establishing trustworthiness among nodes to isolate malicious nodes [12]. Moreover, to build reliable communication for SIoT, Chen et al. proposed an adaptive trust management protocol which adaptively chooses the best trust parameter settings w.r.t. the changing IoT social conditions to assess the trust correctly and maximize the application performance [2]. To the best of our knowledge, how to efficiently utilize the collaboration capability of SIoT to complete the complex tasks remains unexplored.

To complete the complex tasks under SIoT environments, one basic solution is to specify all the required functions of the complex task and perform the required functions on the corresponding SIoT. However, since the number of SIoT objects with the same required functions is tremendous, it is extremely redundant and inefficient to perform the functions on all the compliant SIoT objects. Meanwhile, users also pay the usage cost based on the amount of utilization (pay as you go) in the forms of rental fee or the cost of requested data. Therefore, we adopt the semantic of top-k query to search the optimal group of SIoT objects with the largest success possibility for completing the complex task. Moreover, due to the network reliability of SIoT, it is desirable that each component within the selected group is tightly-coupled or at least not far from each other.

Take Figure 1 as an example. Since the number of catastrophic wildfires has been steadily rising, the government plans to build a wildfire alarm system from the existing SIoT objects. The wildfire alarm prediction task is correlated to accumulative rainfall, temperature, wind speed, and accumulative snowfall according to previous study [6], and each SIoT object can report at least one measurement within an

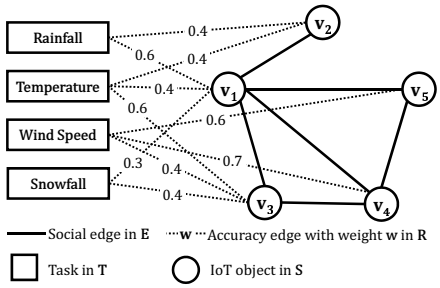


Figure 1: Illustrative example for wildfire detection

accuracy threshold.[1] Therefore, the alarm system issues a top-k query on SIoT and finds the group that maximizes the accuracy of all related measurement. Moreover, under the SIoT environment, each SIoT object replicates its measurement data to its trustworthy "friends" for reliability, fault-tolerance, or accessibility. Therefore, for the data reliability, it is desirable that each component in the selected group is within h-hop from each other or has at least some friends.

Specifically, we propose a new framework, namely, *Task-Optimized Group Search (TOGS)*, to search the best group under the abovementioned SIoT environments. Given a heterogeneous social graph, the set of tasks, the social relationships between SIoT objects, and the relationships between each SIoT object and the task, we propose a new problem family, namely, *Task-Optimized SIoT Selection (TOSS)*, to find the best group of IoT objects for a given set of tasks in the task pool.

To consider different application needs, we propose two different problem formulations for TOSS, namely *Bounded Communication-loss TOSS (BC-TOSS)* and *Robustness Guaranteed TOSS (RG-TOSS)*. While the objective of the two problems are both maximizing the accuracy of performing the given tasks, BC-TOSS aims to bound the communication loss between different SIoT objects, and the goal of RG-TOSS is to provide robustness for message transmission among different SIoT objects. We formulate the problems and prove that they are both NP-Hard and inapproximable within any factor. We propose an error-bounded algorithm with guaranteed performance, namely *Hop-bounded Accuracy-optimized SIoT Extraction (HAE)*, to obtain in polynomial time a solution with objective value no worse than the optimal solution with a bounded error for BC-TOSS. For RG-TOSS, we propose an efficient algorithm to obtain good solutions in polynomial time, namely *Robustness-Aware SIoT Selection (RASS)* which includes effective processing strategies such as *Core-based Robustness Pruning*, *Accuracy-Optimization Pruning*, *Robustness-Guaranteed Pruning*, and *Accuracy-oriented Robustness-aware Ordering*. We conduct a user study for evaluating the effectiveness of the problem formulations, and perform extensive experiments on real datasets to evaluate the proposed algorithms. Experimental results show that our proposed algorithms significantly outperform other baselines. The contributions are summarized as follows.

- For completing complex task in SIoT environment, we propose to model the social edges within SIoT objects with accuracy edges in a heterogeneous graph and propose two different problem formulations, i.e., *BC-TOSS* and *RG-TOSS*, to find suitable SIoT objects. To our best knowledge, there is no real system or existing work in the literature that addresses the issue of group search in SIoT environment.
- We prove that both formulations are NP-Hard and inapproximable within any factors. We then propose a polynomial-time algorithm with performance guarantee and bounded error, namely *Hop-bounded Accuracy-optimized SIoT Extraction (HAE)* for the BC-TOSS problem. We also propose an effective polynomial time algorithm, namely *Robustness-Aware SIoT Selection (RASS)*, to find good solutions for the RG-TOSS problem.
- We conduct a user study on 100 users to validate our two problem formulations. Moreover, we perform extensive experiments on two real datasets. The results show that the proposed algorithms outperform the baselines in terms of objective values and efficiency.

The rest of this paper is organized as follows. Section 2 introduces the related works. Section 3 formulates the problems and proves that the proposed problems are NP-Hard and inapproximable within any factor. Sections 4 and 5 propose algorithms to BC-TOSS and RG-TOSS problems, respectively. Section 6 shows the experimental results and Section 7 concludes this paper.

2. RELATED WORK

A recent line of SIoT research focuses on designing the architectures and protocols for facilitating SIoT under the specific schemes [8, 12, 2]. For example, Nitti et al. propose two trustworthiness management models to suggest strategies of establishing trustworthiness among nodes so that malicious nodes are isolated [12]. Yao et al. propose a joint probabilistic framework for fusing the social relationships between users and IoT objects to improve the accuracy of IoT recommendations [21]. However, these works do not take the capability of collaboration between SIoT into consideration. Moreover, our goal is to find the optimal group of SIoT to accomplish certain tasks, not recommending a single IoT.

To find a cohesive group, many different measurements have been reported in the literature, e.g., diameter [19], density [4, 5], clique and its variations [11]. However, the above works only consider the characteristics inside the group on the existing friendship edges, but *TOGS* needs to consider the accuracy of the assigned task and the "social" tightness among IoT objects. Therefore, new algorithms are necessary to take both types of edges into account. Researches have been proposed to find a socially close group of individuals to invite for activities. In [17], given a query group, the total degree of the community containing this group is maximized. The spatial factor is considered in [10, 20, 23]. Furthermore, the willingness to participate activities is considered in [16]. All the above works keep the social tightness of the target group on the friendship network while maximizing or maintaining some characteristic people care about in an activity. In contrast, in this paper, the accuracy is considered to be maximized so that the query task can find a proper target SIoT group to complete the complex tasks.

On the other hand, expert team formation has attracted a lot of research interests. Forming an expert team is to find

[1]The SIoT objects that cannot report any related measurement can be filtered at the beginning.

Table 1: Notation Summary.

Symbol	Meaning
Q	Query group
$d_S^E(F)$	Largest shortest path distance in F on E
$I_F(t)$	Incident weight of $t \in T$
$deg_H^E(v)$	Inner degree of v in H
$\Omega(F)$	Objective value of target group F
$\alpha(u)$	Sum of incident accuracy edge weight of $u \in S$

a set of experts the required skills, while the communication cost among the chosen experts is minimized so that team members can communicate with each other efficiently. Several communication costs have been proposed under different considerations. For example, Lappas et al. find a team that covers the required skills and minimizes the social diameter of the team or the total edge weight of the spanning tree within the team [9]. Moreover, projects usually require a leader for guiding the direction and negotiation among members. Therefore, Kargar et al. [7] proposed to select a leader for each skill and minimize the social distance from the skill members to each skill leader. In [15], the authors further consider both spatial proximity and skill requirements for finding quick response teams. In contrast, our paper is the first to study the task-optimized group search problem considering SIoT as the input. Based on both social edges and accuracy edges, *TOGS* aims to find the target group with enough social tightness on SIoT to ensure the reliability and while maximizing the accuracy to query tasks.

3. PROBLEM FORMULATIONS

In this paper, we consider a family of *Task-Optimized SIoT Selection (TOSS)* problems on a heterogeneous graph which aim to find the best group of SIoT objects for certain tasks by considering the interactions among different SIoT objects. Specifically, given the heterogeneous graph $G = (T, S, E, R)$, the vertex set T is the *task pool*, i.e., the union of the tasks the SIoT objects can achieve such as measuring humidity, rainfall. Vertex set S represents the set of SIoT objects, where the social relationships among them are captured by the unweighted *social edge* set E, $S \times S \rightarrow E$. Here, a social edge $(u, v) \in E$ represents that SIoT objects u and v can communicate (e.g., using the same communication protocol or equipping the same transmission hardware). For the ease of presentation, we use the term *SIoT Graph*, $G_S = (S, E)$, to describe the graph composed of the SIoT objects set S and the corresponding social edge set E. Finally, R is the set of *task accuracy edge* (*accuracy edge* for short), where each accuracy edge $r = [t, v]$ linking a task vertex $t \in T$ and an SIoT vertex $v \in S$ indicates the accuracy for the SIoT object v to perform task t as the edge weight $w[t, v] \in (0, 1]$. An illustrative example of the heterogeneous graph $G = (T, S, E, R)$ is shown in Figure 1. Table 1 summarizes the notations.

Given the heterogeneous graph $G = (T, S, E, R)$ mentioned above, a query group $Q \subseteq T$ and the desired size p of SIoT objects, the goal of the TOSS problems is to find the group $F \subseteq S$ of exactly p SIoT objects to optimize the accuracy (the definition of *optimizing the accuracy* will be detailed later) of the selected tasks in Q. The size constraint p here represents a budget constraint, i.e., how many SIoT objects we plan to control or carry according to our application scenario. Moreover, based on different practical needs, we apply different constraints on Q to either reduce the communication loss or to increase the robustness of the selected SIoT objects in F. Based on the different constraints, we propose two different problem formulations and algorithm designs.

Specifically, we first propose the *Bounded Communication-loss TOSS (BC-TOSS)* problem by taking into account the communication loss between different SIoT objects in addition to optimizing the accuracy of the selected tasks in Q. To achieve this, we place an upper bound on the hop distance between each pair of SIoT objects in order to limit the number of message forwarding. In other words, the constraint of BC-TOSS is to require the *hop distance* between each pair of vertices in F on E to be at most h, i.e., $d_S^E(F) \leq h$, to reduce the potential communication loss. Please note that since an SIoT object u can forward messages even if it is not selected in F, therefore, the shortest path considered by $d_S^E(F)$ can go through vertices in S but outside F. For example, in Figure 1, if $F = \{v_2, v_3\}$, $d_S^E(F) = 2$ because the shortest path can go through $v_1 \notin F$.

In the second problem, namely *Robustness Guaranteed TOSS (RG-TOSS)*, we pay special attention to the number of different message transmission paths. In other words, RG-TOSS, in addition to optimizing the accuracy of the selected tasks in Q, also requires that each SIoT object in F has at least k neighboring SIoT objects for successfully transmitting the messages. That is, each vertex $v \in F$ must have at least k neighboring vertices also in F.

To measure the solution quality of the returned group F, we consider the sum of the accuracy edge weights incident to each vertex t in Q. Let $I_F(t)$ denote the sum of incident accuracy edge weights of $t \in Q$ to the target group F (*incident weight* of t for short), i.e., $I_F(t) = \sum_{v \in F} w[t, v]$. We then use the sum of incident weights over all tasks t in the task group Q to F to represent the aggregated quality of the returned group F corresponding to Q. In other words, the objective function of the returned group F is defined as $\Omega(F) = \sum_{t \in Q} I_F(t)$. In this paper, we aim to maximize the objective function $\Omega(F)$ to ensure that the tasks in Q are most likely to succeed. Furthermore, we also include an *accuracy constraint* τ in the problem formulation. This accuracy constraint requires that the edge weight of each accuracy edge between Q and F must be at least τ, to ensure the worst case performance of the returned target group.

In the following, we formally formulate the two TOSS problems, namely *Bounded Communication-loss TOSS (BC-TOSS)* and *Robustness Guaranteed TOSS (RG-TOSS)*. We also prove that the proposed two TOSS problems are both NP-Hard and inapproximable within any factors unless P=NP.

3.1 Bounded Communication-loss TOSS (BC-TOSS)

The Bounded Communication-loss TOSS (BC-TOSS) problem is defined as follows.

Problem: Bounded Communication-loss TOSS (BC-TOSS).
Given: Heterogeneous graph $G = (T, S, E, R)$, query group $Q \subseteq T$, hop constraint $h \geq 1$, size constraint $p > 1$, and accuracy constraint $\tau \in [0, 1]$
Objeective: To find a target group $F \subseteq S$ where i) $|F| = p$, ii) $d_S^E(F) \leq h$, and iii) $w[u, v] \geq \tau, \forall u \in Q, v \in F, [u, v] \in R$, such that $\Omega(F) = \sum_{t \in Q} I_F(t)$ is maximized.

Solving the proposed BC-TOSS problem is very challenging due to the interplay of two different edge sets, i.e., groups with largest objective value may not satisfy the hop constraint. In the following, we first analyze the hardness of BC-TOSS by proving that the BC-TOSS problem is NP-Hard. In addition, we prove that there exists no polynomial time approximation algorithm for BC-TOSS. In other words, BC-TOSS is inapproximable within any factor.

THEOREM 1. *BC-TOSS is NP-Hard and inapproximable within any factor.*

PROOF. We prove that BC-TOSS is an NP-Hard problem with the reduction from the $\widehat{p}$-clique problem, which is an NP-Complete problem. Given a graph $G_c = (V_c, E_c)$, where V_c is the set of vertices and E_c is the set of undirected and unweighted edges, and an integer $\widehat{p}$, the decision problem of $\widehat{p}$-clique is to answer whether there exists a subgraph $C_c \subseteq G_c$ such that i) C_c has exactly $\widehat{p}$ vertices, i.e., $|C_c| = \widehat{p}$, and ii) C_c is a complete graph, i.e., $d_S^E(C_c) = 1$, where $d_S^E(C_c)$ is the longest shortest path length on vertex set S and edge set E among all the vertex pairs in C_c.

We transform each instance of $\widehat{p}$-clique to an instance of BC-TOSS as follows. We construct the input graph of BC-TOSS $G = (T, S, E, R)$ by letting $S = V_c$, $E = E_c$, while the task pool T, the accuracy edges in R, the corresponding edge weights, and the query group Q are set arbitrarily. The parameters of BC-TOSS are set as $p = \widehat{p}$, $h = 1$, and $\tau = 0$. In the following, we prove that the decision problem $\widehat{p}$-clique returns TRUE if and only if BC-TOSS has a feasible solution. We first prove the sufficient condition. If $\widehat{p}$-clique returns TRUE with a solution C_c with $d_S^E(C_c) = 1$ and $|C_c| = \widehat{p}$, then C_c must be a feasible solution to BC-TOSS because $d_S^E(C_c) \leq h = 1$ and $|C_c| = \widehat{p} = p$. We then prove the necessary condition. If F is a feasible solution to BC-TOSS, then $d_S^E(F) \leq h = 1$ and $|F| = p$ must hold, which implies that F is also a complete graph of size $\widehat{p} = p$.[2] Therefore, F is also a solution to $\widehat{p}$-clique. This proves that BC-TOSS is NP-Hard.

Finally, we prove hat there exists no approximation algorithm for BC-TOSS unless P=NP. Note that BC-TOSS will return $\Omega(F) = 0$ if $F = \varnothing$, i.e., no feasible solution exists for BC-TOSS. Therefore, if BC-TOSS has a polynomial-time approximation algorithm with an arbitrarily large ratio $\delta < \infty$, the above proof indicates that i) the algorithm is able to obtain a feasible solution to BC-TOSS if $\widehat{p}$-clique returns TRUE, and ii) any BC-TOSS instance with the algorithm returning a feasible solution implies that the corresponding instance in $\widehat{p}$-clique is TRUE. That is, the δ-approximation algorithm can solve $\widehat{p}$-clique in polynomial time, implying that P=NP. Therefore, BC-TOSS has no polynomial-time approximation algorithm unless P=NP. □

Theorem 1 states that the BC-TOSS problem is NP-Hard and inapproximable within any factor. However, we observe that if we slightly relax one constraint of the BC-TOSS problem, we are able to obtain the solution no worse than the optimal solution within polynomial time. We detail this algorithm with performance bound in Section 4.

3.2 Robustness Guaranteed TOSS (RG-TOSS)

[2]Please note that when $d_S^E(F) = 0$, F contains at most 1 vertex, not satisfying the requirement of BC-TOSS which asks $p > 1$.

To find a target group to ensure the communication robustness, i.e., each SIoT object in the target group is able to transmit or backup its data through a number of different neighboring objects, one promising way is to ensure the minimum number of neighbors each SIoT object has in the target group. This motivates us to introduce the degree constraint to guarantee the robustness of communications. We first denote the *inner degree* of a vertex v according to the edge set E in a subgraph $H \subseteq G$ as $deg_H^E(v)$, which is the number of vertices $u \in H$ such that (u, v) is an edge in E. Then, we formally formulate the Robustness Guaranteed TOSS problem as follows, which incorporates the degree constraint in the target group and has the identical objective function and size constraint as the BC-TOSS problem.

Problem: Robustness Guaranteed TOSS (RG-TOSS).
Given: Heterogeneous graph $G = (T, S, E, R)$, query group $Q \subseteq T$, degree constraint $k \geq 1$, size constraint $p > 1$, and accuracy constraint $\tau \in [0, 1]$.
Objeective: To find a target group $F \subseteq S$ where i) $|F| = p$, ii) $deg_F^E(v) \geq k, \forall v \in F$, iii) $w[u, v] \geq \tau, \forall u \in Q, v \in F, [u, v] \in R$, such that $\Omega(F) = \sum_{u \in Q} I_F(u)$ is maximized.

Similar to BC-TOSS, the interplay of the constraints and objective functions on social edges and accuracy edges makes processing RG-TOSS very challenging, especially when the degree constraint of RG-TOSS requires each SIoT object in the returned group to have at least k neighbors in the same group. Please note that this degree constraint requires the *inner degree* to be at least k, which models a more practical situation that we only have control on the selected SIoT objects, i.e., we do not replicate data to or communicate with SIoT objects outside the selected group. Intuitive approaches such as greedily choosing vertices to optimize the objective value does not work because it does not consider the degree constraint and may not obtain feasible solutions. In fact, RG-TOSS is also NP-Hard and inapproximable within any factor, which is proved as follows.

THEOREM 2. *RG-TOSS is NP-Hard and inapproximable within any factors unless P=NP.*

PROOF. We prove that RG-TOSS is NP-Hard with the reduction from an NP-Complete problem, namely $\tilde{k}$-plex problem[14]. Given graph $G = (\tilde{V}, \tilde{E})$ and positive integers $\tilde{p}$ and $\tilde{k}$, the decision problem $\tilde{k}$-plex determines if there exists a set of vertices $C \subseteq \tilde{V}$, such that $deg_{\tilde{C}}^{\tilde{E}}(u) \geq |C| - \tilde{k}, \forall u \in C$ and $|C| = \tilde{p}$.

We transform an instance of the $\tilde{k}$-plex problem to an instance of the RG-TOSS instance by first creating the heterogeneous graph $G = (T, S, E, R)$ with $S = \tilde{V}$ and $E = \tilde{E}$. The task pool T, the set of accuracy edges, and the corresponding accuracy edge weights are set arbitrarily. Moreover, the query group $Q \subseteq T$ is also chosen arbitrarily and $k = \tilde{p} - \tilde{k}$, $p = \tilde{p}$, $\tau = 0$.

We first prove the sufficient condition. If there exists a set $C \subseteq \tilde{V}$ with $deg_{\tilde{C}}^{\tilde{E}}(u) \geq |C| - \tilde{k}, \forall u \in C$ and $|C| = \tilde{p}$, then C must be a feasible solution to the RG-TOSS instance. For the necessary condition, if F is a feasible solution to RG-TOSS, then $|F| = p$ and $deg_{\tilde{C}}^{\tilde{E}}(u) \geq |C| - \tilde{k}, \forall u \in C$ must hold. Therefore, F is also a $\tilde{k}$-plex. This proves that RG-TOSS is NP-Hard.

For the inapproximability, if there exists any δ-approximation

algorithm for any $\delta < \infty$, then such δ-approximation algorithm must be able to obtain a feasible solution of TG-TOSS in polynomial time, which is equivalent to solving the NP-Complete problem $\tilde{k}$-plex in polynomial time. Therefore, RG-TOSS is inapproximable within any factor unless P=NP. The theorem follows. □

Since there exists no approximation algorithm for the RG-TOSS problem unless P=NP, we propose an effective and efficient polynomial-time algorithm to tackle the challenges brought by the interplay of RG-TOSS. We detail the algorithm design in Section 5.

4. ALGORITHM FOR BC-TOSS WITH PERFORMANE GUARANTEE

Theorem 1 in Section 3.1 states that BC-TOSS is NP-Hard and inapproximable within any factor. One simple approach is to enumerate all the combinations to find the optimal solution of BC-TOSS. Due to the large search space, the time complexity of such intuitive approach would be $O(|V|^p)$ which makes it computationally expensive and inapplicable for a large-scale Social IoT network. However, we observe that if we slightly relax the hop constraint, it is possible to find a polynomial time algorithm that can find a solution no worse than the optimal solution (performance guarantee) to BC-TOSS with a *bounded error*. Therefore, in this section, we propose a polynomial time algorithm, namely *Hop-bounded Accuracy-optimized SIoT Extraction (HAE)*, to find the solution with the objective value no worse than the optimal solution while the distance between each pair of vertices on E in the returned group may exceed h, but is guaranteed to be within $2h$. We formally prove the performance guarantee and the error bound of the proposed algorithm.

To avoid generating infeasible solutions, the proposed HAE algorithm first performs a preprocessing step to guarantee that each SIoT object in S has all its incident accuracy edge weights at least τ. That is, this preprocessing step removes each vertex $u \in S$ with an accuracy edge $[u, v]$ for some $v \in Q$ with $w[u, v] < \tau$. Then, the vertices in S which have no incident accuracy edge are also removed because including them in the solution will not increase the objective value.

Afterwards, the HAE algorithm performs a *Sieve Step* to filter out redundant vertices. If an SIoT object $v \in S$ is in the returned group F, then any vertex $u \in F$ must satisfy the following inequality: $d_S^E(u, v) \leq h$. Therefore, the Sieve Step first constructs the candidate set S_v for each SIoT object $v \in S$ where S_v contains only the vertices that are able to form the target group with v, i.e., S_v contains only the vertices within h hops on E from v. Take Figure 1 as an example. Assume $Q = \{$Rainfall, Temperature, Wind Speed, Snowfall$\}$, $p = 3$, $h = 1$, and $\tau = 0.25$. In the Sieve Step, for example, $S_{v_1} = \{v_1, v_2, v_3, v_4, v_5\}$ because all these SIoT objects are within $h = 1$ hop from v_1, and $S_{v_3} = \{v_1, v_3, v_4\}$.

After the Sieve Step is complete, algorithm HAE performs the *Refine Step* to examine each vertex in S_v. Specifically, given SIoT object $u \in S$, we denote $\alpha(u)$ the sum of accuracy edge weights linking from u to the tasks in Q, i.e., $\alpha(u) = \sum_{s \in Q} w[u, s]$. Then, to maximize the objective function, the Refine Step selects p vertices from S_v which have the maximum $\alpha(u)$ and constructs a candidate solution $\mathbb{S}_v$ for v. If $\Omega(\mathbb{S}_v)$ is larger than that of the currently best solution $\mathbb{S}^*$, Algorithm HAE updates $\mathbb{S}^*$ as $\mathbb{S}_v$. Algorithm HAE repeats the examinations of $v \in S$ to construct different candidate solutions, and returns the solution with the maximum objective value as the target group F. Return to our running example in Figure 1. After this step, $\mathbb{S}_{v_1} = \{v_1, v_2, v_3\}$, and $\mathbb{S}_{v_4} = \{v_1, v_3, v_4\}$. Please note that S_{v_2} does not need to be examined because $|S_{v_2}| = 2 < p$, i.e., no feasible solution can be constructed from S_{v_2}. After examining all S_v, the returned target group $F = \{v_1, v_2, v_3\}$, which is the optimal solution.

One major weakness of the steps mentioned above is that algorithm HAE needs to scan over all vertices in S to construct candidate solutions and extract the best one among them. However, this may incur large computation overhead. We observe that if we examine each vertex $v \in S$ in some predefined order, then some $v \in S$ does not need to be examined because the vertices in the corresponding S_v cannot generate a solution better than the best solution obtained so far. Therefore, we propose a vertex-visiting ordering and lookup strategy, namely *Incident Weight Ordering with Top-p Objects Lookup (ITL)* and a powerful pruning strategy, called *Accuracy Pruning (AP)*, to avoid unnecessary search space exploration. ITL visits each vertex $v \in S$ in descending order of $\alpha(v)$, which enables Accuracy Pruning to better estimate the solution quality in each S_v to avoid redundant examinations. Moreover, ITL enables quick candidate solution $\mathbb{S}_v$ generation without sorting the vertices in S_v to extract the top-p vertices with the maximum $\alpha(\cdot)$ values.

Specifically, we associate with each vertex $v \in S$ a list L_v, which is used to store the top-p vertices of the maximum $\alpha(\cdot)$ in S_v. Each time when algorithm HAE examines vertex v and constructs the corresponding S_v in the descending order of $\alpha(v)$, HAE inserts v into each vertex u's list $L_u, \forall u \in S_v$ if $|L_u| < p$. For example in Figure 1, v_3 is visited first because $\alpha(v_3)$ is the largest. After constructing $S_{v_3} = \{v_1, v_3, v_4\}$, HAE also inserts v_3 into L_{v_1}, L_{v_3}, L_{v_4}. The following Lemma 1 proves that the above-mentioned strategy can guarantee that L_u always stores the top-$|L_u|$ vertices with the maximum $\alpha(\cdot)$ in L_u.

LEMMA 1. *For any vertex $u \in S$, its associated L_u stores the top-$|L_u|$ vertices with the maximum $\alpha(\cdot)$ in S_u. Moreover, if $|L_u| < p$, then $\alpha(x) \leq \alpha(u), \forall x \in S_u \backslash L_u$.*

PROOF. HAE visits the vertices $v \in S$ in descending order of $\alpha(v)$. Therefore, for any vertex u, the vertices in its list L_u must be visited before u, leading to $\alpha(x) \geq \alpha(u), \forall x \in L_u$. Moreover, if $u \in S_v$, then $v \in S_u$ as well. Therefore, the vertices in L_u must be in S_u. Since L_u stores at most the first p vertices visited by HAE in S_u, L_u stores the top-$|L_u|$ vertices with the maximum $\alpha(\cdot)$ in S_u.

Please note that the vertices in L_u must have been visited by HAE and $\alpha(y) \geq \alpha(u), \forall y \in L_u$. If $|L_u| < p$, then the vertices in $S_u \backslash L_u$ must not have been visited by HAE yet. According to the vertex-visiting ordering, $\alpha(x) \leq \alpha(u), \forall x \in S_u \backslash L_u$ holds. The lemma follows. □

HAE is equipped with a powerful pruning, namely *Accuracy Pruning*, which can avoid the examination of redundant S_v which never generates better solutions than the currently best solution $\mathbb{S}^*$. Accuracy Pruning works as follows. When Algorithm HAE visits a vertex $v \in S$, before constructing S_v to include the vertices within h hops of v, it first examines the list L_v to check if S_v has a chance to generate a better solution than the currently best solution $\mathbb{S}^*$. If S_v cannot,

HAE skips v and proceeds to examine the next vertex. This saves the computation of traversing the graph to construct S_v. Specifically, the following lemma shows the pruning condition and the correctness of the Accuracy Pruning.

LEMMA 2. ***Accuracy Pruning.*** *Given $v \in S$ and the currently best solution $\mathbb{S}^*$, if $\Omega(L_v)+(p-|L_v|)\alpha(v) \leq \Omega(\mathbb{S}^*)$ holds, S_v can be safely pruned without examination.*

PROOF. Let M_v denote the p vertices with the maximum $\alpha(\cdot)$ values in S_v, and $\widehat{S}_v$ be an arbitrary subset of S_v with $|\widehat{S}_v| = p$. Then $\Omega(M_v) \geq \Omega(\widehat{S}_v)$ must hold. We would like to show that if S_v is pruned by Accuracy Pruning, then there does not exist any $\widehat{S}_v$ such that $\Omega(\widehat{S}_v) > \Omega(\mathbb{S}^*)$.

We prove by contradiction. Assume that $\Omega(M_v) > \Omega(\mathbb{S}^*)$, then $\Omega(M_v) > \Omega(\mathbb{S}^*) \geq \Omega(L_v)+(p-|L_v|)\alpha(v)$ must hold because S_v is pruned by Accuracy Pruning. Case 1) If $|L_v| = p$, then $\Omega(M_v) = \Omega(L_v)$ according to Lemma 1, and we will conclude that $\Omega(M_v) > \Omega(M_v)$ which leads to a contradiction. 2) If $|L_v| < p$, $\sum_{x\in M_v} \alpha(x) > \sum_{x\in L_v} \alpha(x) + (p - |L_v|)\alpha(v)$ holds. According to Lemma 1, $\sum_{x\in(M_v\setminus L_v)} \alpha(x) > (p - |L_v|)$ holds. Therefore, there exists $x \in (M_v \setminus L_v) \subseteq S_v$ L_v such that $\alpha(x) > \alpha(v)$, which contradicts with Lemma 1. Since the above two cases lead to contradictions, $\Omega(M_v) \leq \Omega(\mathbb{S}^*)$ must hold, which leads to $\Omega(\mathbb{S}^*) \geq \Omega(M_v) \geq \Omega(\widehat{S}_v)$. Therefore, if S_v is pruned by Accuracy Pruning, any p-vertex subset $\widehat{S}_v \subseteq S_v$ must have $\Omega(\widehat{S}_v) \leq \Omega(\mathbb{S}^*)$, i.e., S_v cannot generate any solution with objective value better than the currently best solution $\mathbb{S}^*$. The lemma follows. □

Return to our running example in Figure 1. When Algorithm HAE visits v_4, $L_{v_4} = \{v_1, v_3\}$, and the currently best solution $\mathbb{S}^*$ is $\{v_1, v_2, v_3\}$ with $\Omega(\mathbb{S}^*) = 3.5$. In this case, $\Omega(L_{v_4})+(p-|L_{v_4}|)\cdot\alpha(v_4) = 2.7+1\cdot0.7 = 3.4 < \Omega(\mathbb{S}^*) = 3.5$, and Accuracy Pruning prunes v_4. Therefore, Algorithm HAE avoids examining v_4 and does not need to construct S_{v_4} because any subset with p vertices of S_{v_4} will never become a solution better than $\mathbb{S}^*$. The pseudo code of algorithm HAE is shown in Algorithm 1.

In the following, we prove the performance guarantee and error bound of the proposed algorithm. We first prove that, if the optimal solution $\mathbb{S}^{OPT}$ contains a vertex $v \in S$, then $\mathbb{S}^{OPT} \subseteq S_v$ must hold. That is, algorithm HAE does not need to examine any vertex outside S_v if $v \in \mathbb{S}^{OPT}$.

LEMMA 3. *If the optimal solution $\mathbb{S}^{OPT}$ contains vertex $v \in S$, then $\mathbb{S}^{OPT} \subseteq S_v$ holds.*

PROOF. Assume that there exists vertex $v' \in \mathbb{S}^{OPT}$ such that $\mathbb{S}^{OPT}$ is not a subset of $S_{v'}$. Since $\mathbb{S}^{OPT}$ is not a subset of $S_{v'}$, we can find a vertex $u' \in \mathbb{S}^{OPT}$ such that $u' \notin S_{v'}$. In other words, $d_S^E(u', v') > h$ where $d_S^E(u', v') > h$ is the shortest path distance from u' to v' on E. However, from the hop constraint, we know that $d_S^E(u, v') \leq h, \forall u \in \mathbb{S}^{OPT}$ which is a contradiction. Therefore, if $\mathbb{S}^{OPT}$ contains vertex $v \in S$, then $\mathbb{S}^{OPT} \subseteq S_v$ holds. □

We now prove that the proposed HAE algorithm is able to obtain the solution no worse than the optimal solution (performance guarantee) with an error bound h.

THEOREM 3. *The solution F returned by algorithm HAE is no worse than the optimal solution $\mathbb{S}^{OPT}$ to BC-TOSS with an error bound h. That is, $\Omega(F) \geq \Omega(\mathbb{S}^{OPT})$ with $d_S^E(F) \leq 2h$.*

Algorithm 1: Hop-bounded Accuracy-optimized SIoT Extraction (HAE)

Input: $G = (T, S, E, R)$, Q, h, p, τ
Output: F

```
1  begin
2      Remove each u ∈ S where w[u, v] < τ for v ∈ Q
3      S* ← ∅
4      foreach v ∈ S in descending order of α(v) do
5          if v is pruned by Accuracy Pruning then
6              Continue
7          S_v ← { u ∈ S | d_V^E(u, v) ≤ h }
8          if |S_v| < p then
9              Continue
10         if ∃u ∈ S_v with |L_u| < p then
11             Add v into L_u
12         S_v ← {u_1, .., u_p}, which are the p vertices with
               maximum α(u_i) in S_v (extracted with the aid
               from L_v)
13         if Ω(S_v) > Ω(S*) then
14             S* ← S_v
15     F ← S*
16     return F
```

PROOF. Lemma 3 states that if vertex v is included in the optimal solution $\mathbb{S}^{OPT}$, then $\mathbb{S}^{OPT} \subseteq S_v$. Because HAE chooses the p vertices with maximum $\alpha(\cdot)$ in S_v as $\mathbb{S}_v$, there exists no other p-vertex subset of S_v with a larger objective value. Therefore, if $\mathbb{S}^{OPT} \subseteq S_v$, $\Omega(\mathbb{S}_v) \geq \Omega(\mathbb{S}^{OPT})$ must hold. On the other hand, $\Omega(\mathbb{S}^*) \geq \Omega(\mathbb{S}_v), \forall v \in S$, therefore, $\Omega(\mathbb{S}^*) \geq \Omega(\mathbb{S}^{OPT})$ holds. Moreover, Lemma 2 shows that Accuracy Pruning only prunes the examination of S_v if it cannot generate a better solution than $\mathbb{S}^*$. Please note that for any $v \in S$, $d_S^E(S_v) \leq 2h$. Therefore, F returned by algorithm HAE is no worse than the optimal solution with $d_S^E(F) \leq 2h$. The theorem follows. □

THEOREM 4. *HAE has time complexity $O(|R| + |S||E|)$.*

PROOF. HAE removes the SIoT objects that do not satisfy the accuracy constraint in $O(|R|)$ time. Sorting $v \in S$ in descending order of $\alpha(v)$ takes $O(|S|log|S|)$ time. That is, HAE spends $O(|R| + |S|log|S|)$ time for preprocessing.

HAE then considers each v in descending order of $\alpha(v)$. It first takes $O(|S| + |E|)$ time for Accuracy Pruning and extracting S_v for v. Then, HAE takes $O(|V|)$ time to check if there exists $u \in S_v$ with $|L_u| < p$ and $O(|V|)$ time to choose the p vertices u_i with the maximum $\alpha(u_i)$ from S_v. In summary, the time complexity of HAE is $O(|R|+|S|log|S|)+$ $O(|S|(|S| + |E| + |V| + |V|)) = O(|R| + |S||E|)$. □

Although there is a bounded error h for F, in Section 6, we show that most F returned by HAE still satisfy the hop constraint with experiments conducted on real datasets.

5. ALGORITHM DESIGN FOR RG-TOSS

As proved in Section 3.2, RG-TOSS is NP-Hard and inapproximable within any ratio, indicating that RG-TOSS is very challenging due to the interplay of accuracy and communication robustness, i.e., the SIoT objects which have high accuracy may not have robust communications, and

those with robust communication capability may not always have the optimized accuracy of the assigned tasks. To optimize the objective function, one simple approach is to greedily select F containing the p SIoT objects with the largest incident weights. However, this greedy approach may result in a set of SIoT objects that cannot communicate with each other at all. Another approach is to enumerate all the combinations of the SIoT objects. Although this brute-force approach can obtain the optimal solution, it incurs a prohibitively high computation complexity and thus is impractical.

To strike a good balance between solution quality and efficiency, in this section, we propose a polynomial-time algorithm to RG-TOSS, namely *Robustness-Aware SIoT Selection (RASS)* which can obtain good solutions very efficiently. RASS employs a bottom-up approach to construct different partial solutions while considering the accuracy and communication robustness simultaneously. To incrementally construct good partial solutions and lead to good solutions eventually, we propose an effective ordering strategy, called *Accuracy-oriented Robustness-aware Ordering (ARO)*, to prioritize the selections of SIoT objects into partial solutions. Moreover, we also propose effective pruning strategies, namely *Core-based Robustness Pruning (CRP) Accuracy-Optimization Pruning (AOP)*, and *Robustness-Guaranteed Pruning (RGP)*, which are based on our observations in different dimensions to avoid constructing partial solutions that can never grow into better solutions, in order to significantly reduce the computation time of RASS.

Specifically, to significantly reduce the computation time, RASS first employs a filter strategy to remove from G each SIoT object $u \in S$ not satisfying the accuracy constraint. Afterwards, RASS performs *Core-based Robustness Pruning (CRP)* to remove the SIoT objects in S which will not lead to feasible solutions. A $\widehat{k}$-core $C_{\widehat{k}}$ is a graph where each vertex $v \in C_{\widehat{k}}$ has degree at least $\widehat{k}$ [13]. A $\widehat{k}$-core $C_{\widehat{k}}$ is *maximal* if there does not exist another $\widehat{k}$-core that is a superset of $C_{\widehat{k}}$. Maximal $\widehat{k}$-core can be obtained in polynomial time.[3] In Core-based Robustness Pruning, RASS extracts the maximal k-core C_k from the graph formed by the SIoT objects and the corresponding social edge set, i.e., $G_S = (S, E)$, where k is the degree constraint. RASS then trims the SIoT objects from S which are not included in the maximal k-core C_k. The following lemma shows that the SIoT objects in $S \backslash C_k$ can be safely trimmed.

LEMMA 4. ***Core-based Robustness Pruning.*** *Given maximal k-core $C_k \subseteq G_S$ and any feasible solution F to RG-TOSS, $(S \backslash C_k) \cap F = \varnothing$ must hold, indicating that SIoT objects in $S \backslash C_k$ can be safely trimmed.*

PROOF. Suppose v is an SIoT object which is not included in the maximal k-core C_k, i.e., $v \in S \backslash C_k$. We prove this lemma by contradiction. Assume that F is a feasible solution and $v \in F$. As F is a feasible solution, $deg_F^E(u) \geq k, \forall u \in F$ must hold. Therefore, F is a k-core and $F \subseteq C_k$ holds (according to the definition of maximal k-core). Since $v \in F$, $v \in C_k$ must hold, which leads to a contradiction. Therefore, $v \notin F$ and v can be safely trimmed, $\forall v \in S \backslash C_k$. The lemma follows. □

[3]Please note that the maximal $\widehat{k}$-core may contain multiple connected components.

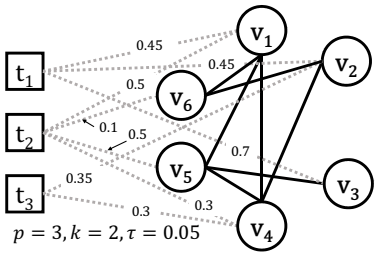


Figure 2: Running example of RG-TOSS

Consider the running example in Figure 2. Given the heterogeneous graph G with $p = 3$, $k = 2$, and $\tau = 0.05$. Since the maximal 2-core in $G_S = (S, E)$ is $\{v_1, v_2, v_4, v_5, v_6\}$, Core-based Robustness Pruning removes v_3 from S because v_3 will never be included in any feasible solution.

In algorithm RASS, each partial solution σ_i is defined as $\sigma_i = \{\mathbb{S}_i, \mathbb{C}_i\}$ where $\mathbb{S}_i$ is the solution set containing a set of SIoT objects, and $\mathbb{C}_i$ denotes the set of candidate SIoT objects that can be considered by the current partial solution σ_i. During the process of RASS, RASS maintains a priority queue $\mathbb{U}$ to store different partial solutions. Let $S = \{v_1, ..., v_{|S|}\}$. In the very beginning, RASS generates $|S|$ initial partial solutions and pushes them into $\mathbb{U}$, where each partial solution contains $\{\{v_i\}, \bigcup_{j \in [i+1, |S|]} v_j\}$ for each different $v_i \in S$. Return to the running example in Figure 2. In the beginning, priority queue $\mathbb{U}$ contains the following partial solutions: $\{\mathbb{S}_1 = \{v_1\}, \mathbb{C}_1 = \{v_2, v_4, v_5, v_6\}\}$,$\{\mathbb{S}_2 = \{v_2\}, \mathbb{C}_2 = \{v_4, v_5, v_6\}\}$,$\{\mathbb{S}_3 = \{v_4\}, \mathbb{C}_3 = \{v_5, v_6\}\}$. Please note that v_3 does not appear because it has been pruned by Core-based Robustness Pruning. Moreover, there is no partial solution $\{\mathbb{S}_4 = \{v_5\}, \mathbb{C}_4 = \{v_6\}\}$ because $|\mathbb{S}_4 \cup \mathbb{C}_4| < p = 3$. That is, even if we move all the candidate SIoT objects in $\mathbb{C}_4$ into $\mathbb{S}_4$, we still cannot form a feasible solution with exactly p SIoT objects. Similarly, $\{\mathbb{S}_5 = \{v_6\}, \mathbb{C}_5 = \phi\}$ does not exist in $\mathbb{U}$ as well.

At each step afterwards, RASS generates a new partial solution $\sigma' = \{\mathbb{S}', \mathbb{C}'\}$ as follows. RASS first pops from the priority queue $\mathbb{U}$ a partial solution $\sigma = \{\mathbb{S}, \mathbb{C}\}$ based on Accuracy-oriented Robustness-aware Ordering (ARO, will be detailed later), and RASS creates a copy of σ, i.e., σ'. Then for σ', RASS moves an SIoT object u with the maximum $\alpha(u)$ from its candidate SIoT object set $\mathbb{C}'$ into its solution set $\mathbb{S}'$. Therefore, σ' becomes a new partial solution, i.e., $\mathbb{S}' \backslash \mathbb{S} = \{u\}$.

Return to the running example in Figure 2, after initialization, assume ARO in RASS selects the partial solution $\sigma = \{\{v_1\}, \{v_2, v_4, v_5, v_6\}\}$ for expansion. RASS first creates a copy of σ, i.e., $\sigma' = \{\mathbb{S}' = \{v_1\}, \mathbb{C}' = \{v_2, v_4, v_5, v_6\}\}$. Since choosing v_2 for expanding $\mathbb{S}'$ does not satisfy ARO, RASS choose v_4 which satisfies ARO and has the maximum $\alpha(\cdot)$. Therefore, v_4 is moved to $\mathbb{S}'$ and $\sigma' = \{\{v_1, v_4\}, \{v_2, v_5, v_6\}\}$ is a new partial solution. RASS then removes v_4 from $\mathbb{C}$ of σ to avoid generating duplicate partial solution as σ' in the future, and pushes $\sigma = \{\{v_1\}, \{v_2, v_5, v_6\}\}$ back to the priority queue $\mathbb{U}$. Please note that, σ is always inserted back into the priority queue (unless $|\mathbb{S}| + |\mathbb{C}| < p$) because σ is able to generate other new partial solutions by expanding its solution set with other vertices. Moreover, in order to guarantee not generating duplicate partial solutions, the SIoT object that is moved from $\mathbb{C}'$ to $\mathbb{S}'$ is removed from $\mathbb{C}$ of σ. For example, v_4 is removed from $\mathbb{C}$ when σ is pushed back into $\mathbb{U}$.

If the solution set $\mathbb{S}' \in \sigma'$ contains p SIoT objects, satisfies the degree constraint, and $\Omega(\mathbb{S}')$ is larger than the currently best solution $\mathbb{S}^*$, RASS updates $\mathbb{S}^*$ as $\mathbb{S}'$. If $\mathbb{S}'$ contains fewer than p SIoT objects, RASS inserts σ' into the priority queue $\mathbb{U}$. In the second round in our running example, RASS pops $\sigma = \{\{v_1, v_4\}, \{v_2, v_5, v_6\}\}$ according to ARO, and generates a new partial solution $\sigma' = \{\{v_1, v_4, v_5\}, \{v_2, v_6\}\}$. Since $\mathbb{S}' = \{v_1, v_4, v_5\}$ contains $p = 3$ SIoT objects and satisfies the constraints, and $\mathbb{S}'$ is the first feasible solution obtained so far, RASS sets $\mathbb{S}^* = \mathbb{S}'$. RASS pushes the original σ back to $\mathbb{U}$ (σ' does not have to be pushed back because $|\mathbb{S}'| = 3$). The number of expansions on partial solutions RASS performs is bounded by a parameter λ. After λ expansions of partial solutions, RASS outputs the best solution $\mathbb{S}^*$ as the solution. The setting of λ represents a trade-off between efficiency and solution quality. We will compare the performance of RASS under different λ values in the experimental results in Section 6.

In the following, we detail the important strategies employed in our framework. These strategies include *Accuracy-oriented Robustness-aware Ordering*, *Accuracy-Optimization Pruning* and *Robustness-Guaranteed Pruning*, which can significantly improve the efficiency and solution quality of the proposed RASS algorithm. The pseudo code of algorithm RASS is detailed in Algorithm 2.

Algorithm 2: Robustness-Aware SIoT Selection (RASS)

Input: $G = (T, S, E, R)$, Q, k, p, τ, λ
Output: F

```
1  begin
2      Remove each u ∈ S where w[u, v] < τ for v ∈ Q
3      S* ← ∅; U ← ∅; expand ← 0
4      CRP (Lemma 4) on G_S = (S, E)
5      foreach v_i ∈ S = {v_1, .., v_|S|} do
6          push {{v_i}, ⋃_{j∈[i+1,|S|]} v_j} into U
7      while expand < λ do
8          expand ← expand + 1
9          Pop σ = {S, C} from U based on ARO
10         if σ can be pruned by AOP (Lemma 5) or RGP
             (Lemma 6) then
11             Continue
12         Copy σ to σ'; Push σ back into U if |S| + |C| ≥ p
13         u ← argmax_{x∈C'} α(x)
14         Move u from C' to S'
15         if |S'| = p and Ω(S') > Ω(S*) then
16             S* ← S'
17         else if |S'| < p then
18             push σ' back to U
19     F ← S*
20     return F
```

5.1 Accuracy-oriented Robustness-aware Ordering

The selection of partial solution σ from the priority queue to construct σ' is critical to the solution quality and algorithm efficiency. This is because a carefully selected σ can generate a good solution earlier, which can be used to prune other partial solutions afterwards. One simple approach, called *Accuracy Ordering*, is to select σ where its corresponding solution set $\mathbb{S}$ has the maximum $\Omega(\mathbb{S})$ (i.e., maximum accuracy), to expand into σ'.

Consider Figure 2, after initialization, Accuracy Ordering would select $\{\{v_1\}, \{v_2, v_4, v_5, v_6\}\}$ because $\{v_1\}$ has the maximum $\Omega(\mathbb{S})$. Moreover, this partial solution is copied and expanded into $\{\{v_1, v_2\}, \{v_4, v_5, v_6\}\}$ because v_2 has the maximum $\alpha(\cdot)$ in $\{v_2, v_4, v_5, v_6\}$. However, $\{v_1, v_2\}$ will not become a feasible solution because $p = 3$ and $k = 2$, i.e., requiring the SIoT objects in the solution to connect to each other. This example demonstrates that Accuracy Ordering does not consider the degree constraint and is inclined to obtain a set of SIoT objects without communication robustness, resulting in the generation of a large number of infeasible solutions.

To tackle the problem of Accuracy Ordering, we propose *Accuracy-oriented Robustness-aware Ordering (ARO)* to consider both accuracy and communication robustness simultaneously. The idea of ARO is to prioritize the selection of Accuracy Ordering with additional conditions of the communication robustness. Recall that Accuracy Ordering pops the partial solution $\sigma = \{\mathbb{S}, \mathbb{C}\}$ with the maximum $\Omega(\mathbb{S})$ from $\mathbb{U}$. Then σ' is constructed by moving the vertex $u \in \mathbb{C}$ which incurs the maximum $\alpha(u)$ to $\mathbb{S}$. In ARO, σ is selected from the priority queue only when $(\mathbb{S} \cup \{u\})$ (u incurs the maximum $\alpha(u)$ in $\mathbb{C}$) has sufficient communication robustness. In this case, ARO effectively guides RASS to expand good partial solutions which has high potential to satisfy the degree constraint.

Specifically, given a partial solution $\sigma = \{\mathbb{S}, \mathbb{C}\}$, let $\Delta(\mathbb{S}) = \frac{\sum_{v \in \mathbb{S}} deg_{\mathbb{S}}^{E}(v)}{|\mathbb{S}|}$ be the average inner degree of $\mathbb{S}$, and let u be the SIoT object in $\mathbb{C}$ which incurs the $\alpha(u)$. In ARO, we first assume that u is added to $\mathbb{S}$. Then, we examine if the communication robustness of the new set $\mathbb{S} \cup \{u\}$ is sufficiently large. After that, from those partial solutions that satisfy the communication robustness requirement, we select and pop the partial solution which incurs the maximum $\Omega(\cdot)$ for expansion. The communication robustness of $\mathbb{S} \cup \{u\}$ is considered sufficiently large if the following *Inner Degree Condition (IDC)* holds: $\Delta(\mathbb{S} \cup \{u\}) \geq |\mathbb{S} \cup \{u\}| - \frac{\mu \cdot |\mathbb{S} \cup \{u\}| + p - 1}{p - 1}$, where μ is a self-adjusting filtering parameter.

The filtering parameter μ is important for finding good feasible solutions quickly. Specifically, μ is set as $p - k - 1$ initially to provide a more strict filtering power when selecting SIoT objects into $\mathbb{S}$ to fulfill the degree constraint, i.e., when μ is larger, vertex u passes IDC when u has more inner degree in the current $\mathbb{S} \cup \{u\}$. When no SIoT object satisfies IDC with the current μ values, ARO decreases μ to lower the threshold until at least one vertex u satisfies IDC.

In our example shown in Figure 2, given $\sigma = \{\mathbb{S} = \{v_1\}, \mathbb{C} = \{v_2, v_4, v_5, v_6\}\}$, with $\mu = p - k - 1 = 0$, ARO avoids to select $v_2 \in \mathbb{C}$ to expand $\mathbb{S}$ (which would be chosen by the intuitive Accuracy Ordering) because $\Delta(v_1 \cup \{v_2\}) < 2 - 1$ does not satisfy the Inner Degree Condition. In fact, selecting v_2 by Accuracy Ordering would not expand σ into any feasible solution, but only costs unnecessary expansions. Instead, ARO considers the set of SIoT objects in $\mathbb{C}$ which satisfies Inner Degree Condition, i.e., $\{v_4, v_5, v_6\}$, and picks v_4 because v_4 incurs the maximum $\alpha(\cdot)$. As a result, ARO expands σ to $\sigma' = \{\{v_1, v_4\}, \{v_2, v_5, v_6\}\}$.

5.2 Pruning Strategies

The ordering strategy, i.e., ARO, described in Section 5.1 prioritizes the expansion of partial solutions that are likely to become good feasible solutions. It is expected that ARO is able to obtain the first feasible solution which follows the degree constraint much earlier than Accuracy Ordering because inner degrees are examined during the process. To reduce the examination of unnecessary partial solutions which will never become better feasible solutions, we further derive two pruning strategies, namely *Accuracy-Optimization Pruning (AOP)* and *Robustness-Guaranteed Pruning (RGP)*.

Accuracy-Optimization Pruning (AOP) keeps tracks of the objective value of the currently best solution, i.e., $\Omega(\mathbb{S}^*)$ and removes the partial solutions that will never become a better solution than $\mathbb{S}^*$ by deriving the upper bound on the objective values of the partial solutions. Equipped with AOP, RASS is able to avoid unnecessary expansions of partial solutions and significantly reduces the computation time. On the other hand, Robustness-Guaranteed Pruning (RGP) considers the communication robustness of the partial solutions. RGP prunes the partial solutions (discards it from $\mathbb{U}$ directly) if they cannot grow into feasible solutions, i.e., satisfying the degree constraint. With RGP, algorithm RASS can avoid spending unnecessary computation resource on partial solutions that will not become feasible solutions.

Specifically, given the currently best solution $\mathbb{S}^*$ and its accuracy $\Omega(\mathbb{S}^*)$, Lemma 5 states Accuracy-Optimization Pruning.

LEMMA 5. ***Accuracy-Optimization Pruning (AOP).*** *Partial solution $\sigma = \{\mathbb{S}, \mathbb{C}\}$ can be removed from the priority queue $\mathbb{U}$ if $\sum_{v\in\mathbb{S}} \alpha(v) + (p-|\mathbb{S}|)\cdot \max_{u\in\mathbb{C}} \alpha(u) \leq \Omega(\mathbb{S}^*)$ holds.*

PROOF. The first term of the inequality is the total accuracy of the SIoT objects in $\mathbb{S}$, and the second term: $(p - |\mathbb{S}|) \cdot \max_{u\in\mathbb{C}} \alpha(u)$ is an upper bound on the total accuracy the current partial solution can achieve by adding $(p - |\mathbb{S}|)$ SIoT objects. Therefore, if the inequality holds, any solution constructed from the current partial solution σ will never have total accuracy larger than the currently best solution $\mathbb{S}^*$ and thus can be safely pruned. □

Return to the running example in Figure 2. After obtaining $\mathbb{S}^* = \{v_1, v_4, v_5\}$, assume that RASS is considering to expand $\sigma = \{\mathbb{S} = \{v_2\}, \mathbb{C} = \{v_4, v_5, v_6\}\}$. Since $\sum_{v\in\mathbb{S}} \alpha(v) = 0.8$ and $(p - |\mathbb{S}|) \cdot \max_{u\in\mathbb{C}} \alpha(u) = 2 \cdot 0.6$, $\sum_{v\in\mathbb{S}} \alpha(v) + (p - |\mathbb{S}|) \cdot \max_{u\in\mathbb{C}} \alpha(u) = 2.0 < \Omega(\mathbb{S}^*) = 2.05$. Therefore, σ can be directly removed from $\mathbb{U}$ without any further expansions.

On the other hand, Robustness-Guaranteed Pruning considers the degrees of the SIoT objects inside $\mathbb{S}$ and those outside $\mathbb{S}$ of a partial solution. The following Lemma 6 details RGP.

LEMMA 6. ***Robustness-Guaranteed Pruning (RGP).*** *Partial solution $\sigma = \{\mathbb{S}, \mathbb{C}\}$ can be removed from the priority queue $\mathbb{U}$ if one of the conditions holds: 1) $p - |\mathbb{S}| + \min_{v\in\mathbb{S}} deg^E_{\mathbb{S}}(v) < k$, or 2) $\sum_{v\in\mathbb{C}} deg^E_{\mathbb{C}\cup\mathbb{S}}(v) < k(p - |\mathbb{S}|)$.*

PROOF. For the first condition, $p - |\mathbb{S}|$ is the number of SIoT objects which will be added into $\mathbb{S}$, and $\min_{v\in\mathbb{S}} deg^E_{\mathbb{S}}(v)$ is the minimum inner degree of the SIoT objects in $\mathbb{S}$. Therefore, the left-hand-side of the first condition is the upper bound on the inner degree of the vertex with the minimum inner degree in $\mathbb{S}$. If the first condition holds, at least one SIoT object will not satisfy the degree constraint afterwards.

The first condition considers the SIoT objects that have been moved into $\mathbb{S}$. For the second condition, it considers the SIoT objects that are in $\mathbb{C}$. For the Right-Hand-Side of the inequality, $(p - |\mathbb{S}|)$ is the number of SIoT objects that need to be moved from $\mathbb{C}$ into $\mathbb{S}$, and $k(p-|\mathbb{S}|)$ is the number of total vertex degrees the added vertices should have to become a feasible solution. Therefore, if the $\sum_{v\in\mathbb{C}} deg^E_{\mathbb{C}\cup\mathbb{S}}(v)$, i.e., the total vertex degrees $\mathbb{C}$ can provide, is fewer than $k(p - |\mathbb{S}|)$, the partial solution σ will never grow into a feasible solution. □

Return to Figure 2, assume RASS is now examining $\sigma = \{\{v_2\}, \{v_4, v_5, v_6\}\}$. Since $\sum_{v\in\mathbb{C}} deg^E_{\mathbb{C}}(v) = 1 + 1 + 0$, which is smaller than $k(p-|\mathbb{S}|) = 2\cdot(3-1)$. Therefore, σ can be directly removed from $\mathbb{U}$ without further expansions. The following Theorem 5 summarizes the time complexity of RASS.

THEOREM 5. *RASS has time complexity $O(|R| + \lambda(|S| + \lambda)p^2)$.*

PROOF. RASS removes vertices which do not satisfy the accuracy constraint from S in $O(|R|)$ time. Core-based Robustness Pruning is performed in $O(|S|)$, and for initialization, RASS spends $O(|S|)$ to construct and push each partial solution $\{\{v_i\}, \bigcup_{j\in[i+1,|S|]} v_j\}$ into $\mathbb{U}$. Therefore, before RASS expands any partial solution, it takes $O(|R| + |S|)$ time.

To identify and pop a partial solution, since $\mathbb{U}$ has at most $(|S| + \lambda)$ partial solutions, RASS performs $O(|S| + \lambda)$ times Inner Degree Condition verification (which takes $O(p^2)$). That is, it costs $O((|S| + \lambda)p^2)$ to identify and pop the partial solution σ. It takes $O(p)$ and $O(|S|)$ time to examine Accuracy-Optimization Pruning and Robustness-Guaranteed Pruning, respectively, and $O(|S|)$ time to copy σ to σ'. Therefore, expanding a partial solution takes $O((|S|+\lambda)p^2)$ time. Since RASS expands at most λ partial solutions, expanding λ partial solutions takes $O(\lambda(|S| + \lambda)p^2)$. In summary, the time complexity of RASS is $O(|R|+\lambda(|S|+\lambda)p^2)$. □

6. EXPERIMENT

In this section, we first detail the preparation of the datasets used in our evaluation. Afterwards, we evaluate the performance of the proposed algorithms with real datasets. Since *BC-TOSS* and *BC-TOSS* are NP-hard, we first enumerate all the possible combinations on a small-scale dataset to derive the optimal solution, and compare it with our solutions. Moreover, to evaluate the efficiency and effectiveness of the proposed algorithms, a co-author network is transformed into an SIoT network, where each node in the network contains a skill set (detailed in Section 6.1). Finally, a user study with 100 people is conducted to compare manual coordination with the proposed *HAE* and *RASS*.

6.1 Experiment Setting

The first dataset, *RescueTeams*, contains a small set of the rescue and disaster response teams in Canada[4] and California, USA,[5] with 68 and 77 teams, respectively. We regard

[4]A part of the rescue and disaster response teams can be found on http://en.wikipedia.org/wiki/Canadian_Forces_Search_and_Rescue

[5]A part of the rescue and disaster response teams can be found on http://www.calema.ca.gov/Pages/default.aspx

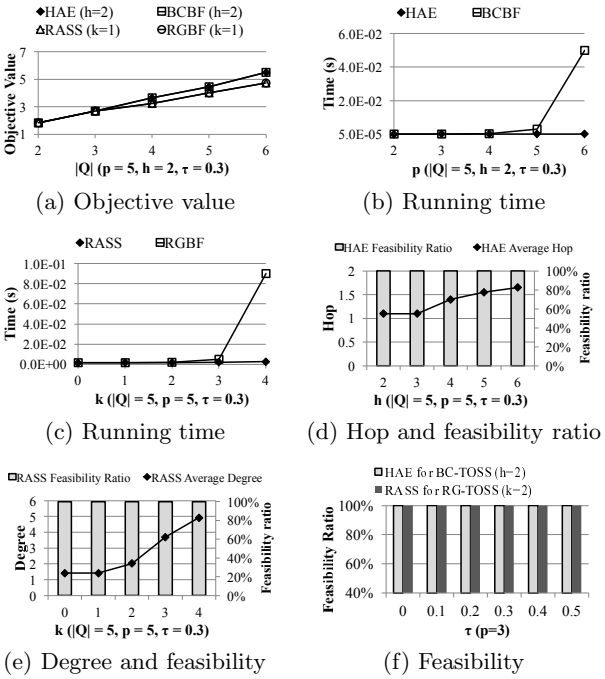
each team as a candidate SIoT object with possession of certain equipment representing proficiency in associated, e.g., a rescue and disaster response team with equipment A and B is viewed as a node in G with skills A and B. Moreover, the accuracies of edges are generated by uniform distribution ranged from 0 to 1. We also collect and analyze the spatial coordinates and characteristics of 34 and 32 disasters occurring in Canada and California, respectively, during the past 5 years to serve as the basis of queries and required skills in our evaluation. The types of disasters include wildfires, hurricanes, floods, earthquakes, and landslides. Due to the lack of social relations in the RescueTeam dataset, we create social links to the dataset based on the distance between two teams. We first sort all the pairwise distances in ascending order and select the top 50% to generate social edges.

Moreover, since there is no public large-scale SIoT dataset with specified tasks, to generate the input for the TOSS problem, we take Dataset *DBLP*, which contains $511,163$ nodes and $1,871,070$ edges, and only entries corresponding to the papers published in the areas of Database (DB), Artificial intelligence (AI), Data mining (DM), and Theory (T) conferences are kept. Only the authors who have at least three papers in the four areas are included, and each author is regarded as an SIoT object and the skills of authors are regarded as the tasks can be assigned to them. Moreover, we generate the skill set and social edges similar to [9]. Specifically, an author owns a skill (terms) if the term appears in at least two titles of papers that he has co-authored. We further generate the accuracy edges of author v_i by first counting the number of times each term appearing in titles of papers that he has co-authored and then normalizing it with the largest counts among all authors. Finally, two authors v_i and v_j are connected if they appear as co-authors in at least two papers in DBLP.

In the following, we compare *HAE* and *RASS* with two baselines. The first baseline is a brute-force method which enumerates all the feasible solutions for *BC-TOSS* (namely *BCBF*) and *RG-TOSS* (namely *RGBF*) to show the difference between the solutions derived from the proposed methods and optimal solutions. Moreover, we compare *HAE* and *RASS* with DpS [4]. DpS is an $O(|V|^{1/3})$-approximation algorithm for finding a p-vertex subgraph $H \subseteq S$ with the maximum density (the number of edges induced by H divided by $|H|$) on E without considering the query group, accuracy edges, hop or degree constraint. Finally, we implement the proposed algorithms, *HAE* and *RASS*, and invite 100 people from various communities, e.g., government, banks, hospitals, technology companies, schools, and businesses to join our user study, to compare the objective values and the time for answering *BC-TOSS* and *RG-TOSS* with manual coordination and proposed algorithms (i.e., *HAE* and *RASS*) to demonstrate the advantages of automatic query answering on SIoT. Each user is asked to plan 20 SIoT object selections for query answering with the query tasks. For the target graph, we sample a topology from Dataset *RescueTeams* and randomly connect edges to the query task with the weighting following the uniform distribution. All the experiments are implemented in an HP DL580 server with 4 Intel Xeon E7-4870 2.4 GHz CPUs and 1 TB RAM.

6.2 Performance Evaluation

6.2.1 RescueTeams



(a) Objective value (b) Running time

(c) Running time (d) Hop and feasibility ratio

(e) Degree and feasibility (f) Feasibility

Figure 3: Experiment results on *RescueTeams* dataset

In the following, we first compare the performance of *HAE* and *RASS* with baselines (*BCBF* and *RGBF*) on Dataset *RescueTeams*. *BCBF* and *RGBF* are brute-force algorithms which enumerate all the combinations of solutions, check the feasibility, and output the feasible solutions with the largest objective value. We randomly sample the query tasks 100 times and report the averaged results.

Figure 3(a) compares the objective values of *HAE* and *RASS* with *BCBF* and *RGBF*, respectively, for different query task sizes $|Q|$, where the budget constraint $p = 5$, hop constraint $h = 2$, and accuracy constraint $\tau = 0.3$. The results show that the objective value of the target group is proportional to the query group size $|Q|$. Moreover, *HAE* and *RASS* always derive the optimal objective value as $|Q|$ grows. The objective values of *HAE* are slightly larger than those of *RASS* since the constraint is looser and thus reduces the solution space. Figure 3(b) presents the running time for answering *BC-TOSS* with different budget constraints p. As p grows, the running time of *BCBF* significantly increases due to the large number of possible combinations, while the running time of *HAE* only slightly increases. On the other hand, Figure 3(c) presents the running time for answering *RG-TOSS* with different degree constraints k. *RASS* significantly outperforms *RGBF* as $|Q|$ grows.

In addition to objective values and running time, the feasibility ratio and average hop of *HAE* are reported in Figure 3(d). Although *HAE* slightly relaxes the hop constraint to derive the optimal solution with a bounded error, all the feasibility ratios w.r.t. different h are 100%. The average hop of solutions derived by *HAE* slightly increases as h grows, which implies that *HAE* finds optimal solutions of which SIoT objects are not far away from each other for providing data and transmission reliability. On the other hand, Figure 3(e) shows the feasibility ratio and average degree of *RASS*. All the feasibility ratios w.r.t. different K are 100%. Note

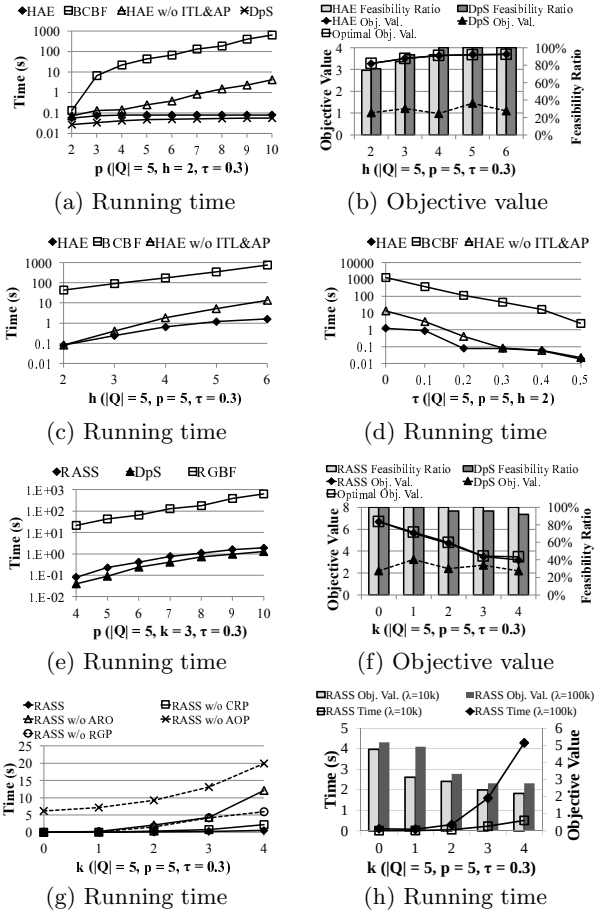


Figure 4: Experiment results on *DBLP* datasets

that the average degree of optimal solutions when $k = 0$ (no degree constraint) and $k = 1$ are close. This is because the rescue teams with different skills are usually not far from each other to cover all the rescue tasks within an area. As such, the average degree of the optimal solution without the degree constraint is more than 1, i.e., nodes are connected instead of being isolated. Figure 3(f) shows the feasibility ratio with different accuracy constraints from 0 to 0.5, and the results indicate the robustness of *HAE* and *RASS* since the feasibility ratios are all 100% given different accuracy constraints τ.

6.2.2 DBLP

We further evaluate and analyze the performance of the proposed algorithms on *DBLP* dataset. Figures 4(a)-(d) show the results for answering *BC-TOSS*. We report the results of *HAE* with three baselines: 1) brute-force method (*BCBF*), 2) Densest p-Subgraph (*DpS*) [4], and 3) *HAE* without Incident Weight Ordering with Top-p Object Lookup and Accuracy Pruning (*HAE w/o ITL&AP*).

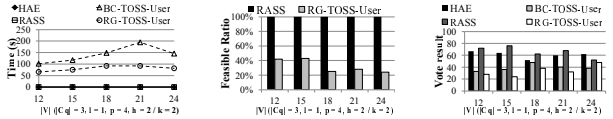
Figure 4(a) shows the running time with different budget constraint p, where $|Q| = 5$, $h = 2$, and $\tau = 0.3$. The result indicates that the running time of *HAE* is close to that of *DpS* but outperforms other baselines. The running time of *DpS* is the smallest since *DpS* only finds the densest p-subgraph without computing the feasibility of solutions. On the other hand, as p grows, the running time of *HAE* is much less than that of *HAE w/o ITL&AP*, which indicates the effectiveness of the lookup and pruning strategy. Figure 4(b) shows the objective values and feasibility ratios with different hop constraints given accuracy constraint $\tau = 0.3$. *DpS* slightly outperforms *HAE* in terms of feasibility ratio since *DpS* finds socially-tight groups which are inclined to satisfy the hop constraint. However, without considering the objective values, the objective values of *DpS* are much smaller than that of *HAE*, while the objective values of *HAE* are close to optimal. Figure 4(c) reports the running time with different hop constraints h. As h increases, the running time of all methods grows linearly, while the running time of *HAE* is still close to 1 second given $h = 6$. We further investigate the relationship between accuracy constraint τ and running time. The results manifest that the running time can be reduced when τ is large since the solution space is significantly reduced with large τ. However, if we set τ with a value close to 1, the solution space may become empty without any feasible solutions.

Figures 4(e)-(h) present the results for answering *RG-TOSS*. The results of *RASS* are compared with those of the brute-force method (*RGBF*) and *DpS*. Given $|Q| = 5$, $k = 3$, and $\tau = 0.3$, Figures 4(e) shows the running time with different budget constraints p. The results indicate that the proposed *RASS* outperforms *RGBF* by at least two orders. On the other hand, Figure 4(f) shows the objective values and feasibility ratios with different k. When the degree constraint k increases, the feasibility ratio of *RASS* is still 100% and outperforms *DpS* since ARO prioritizes the examination of partial solutions that will lead to feasible solutions. Note that *DpS* finds the densest subgraph but may not satisfy the degree constraint since most of the edges in the group may only be incident to some nodes, while the remaining nodes do not satisfy degree constraint. Meanwhile, the objective values of *RASS* are close to those of the optimal solutions.

We further conduct experiments on the running time and objective values with different k as shown in Figure 4(g). As the degree constraint becomes strict, i.e., the requirement of reliability in data transmission becomes high, the objective values become small since the cohesive requirement reduces the number of possible solutions and may not answer the task correctly. Moreover, as k increases, the running time of *RASS* also grows since the complexity of finding a cohesive group is high, e.g., cliques. Figure 4(h) shows the running time of *RASS*, *RASS* without Accuracy-oriented Robustness-aware Ordering (*RASS* w/o ARO), *RASS* without Core-based Robustness Pruning (*RASS* w/o CRP), *RASS* without Accuracy-Optimization Pruning (*RASS* w/o AOP), and *RASS* without Robustness-Guaranteed Pruning (*RASS* w/o RGP). The result manifests that Accuracy-Optimization Pruning (AOP) is the most effective because AOP precisely estimates the upper bounds of partial solutions and effectively prunes the partial solutions that cannot grow into better solutions.

6.2.3 User Study

Here we conduct a user study to show that human computation for *BC-TOSS* and *RG-TOSS* is time-consuming, while the objective values are not close to optimal even when the number of SIoT objects is small. Each user is assigned to solve *BC-TOSS* and *RG-TOSS* on 5 small SIoT networks with vertex set sizes $12, 15, 18, 21$, and 24. To avoid confusing users with the complicated network structure, every vertex is labelled with an objective value, which is the sum-



(a) Running time (b) Feasible ratio (c) Vote result

Figure 5: User Study results

mation of the accuracy edge weights for the assigned tasks. For each instance, the query group size, i.e., $|Q|$, is 3, the budget constraint p is 4, the hop constraint is 2, and the degree constraint is 2.

Figure 5(a) compares manual coordination, *HAE*, and *RASS* in terms of running time. The result indicates that users spend from 50 to 200 seconds solving the *BC-TOSS* and *RG-TOSS* problem, while the running time for *HAE*, and *RASS* is close to 0. Moreover, the time of manual coordination for *BC-TOSS* is greater than that of *RG-TOSS* with different network sizes. However, as shown in Figure 5(b), the feasible ratio of manual coordination for*RG-TOSS* is small, especially for large network size, because it is difficult for users to check the degree constraint on network topology, while maximizing the summation of accuracy. The interplay between network topology and accuracy complicates *RG-TOSS*.

We also ask users to vote for the results of manual coordination, *HAE*, and *RASS*, as shown in Figure 5(c). The result manifests that users think our solutions are better as compared to the solutions found by themselves. Moreover, users think that *RASS* is more helpful since the feasibility examination on network topology for *RG-TOSS* problem is difficult. Therefore, it is desirable to deploy *HAE* and *RASS* as a service for automatic task-optimized group search, especially to address the need of a large group in a massive SIoT network nowadays.

7. CONCLUSIONS

In this paper, we propose and study a family of *Task-Optimized SIoT Selection (TOSS)* problems. To our best knowledge, this is the first paper that considers simultaneously the accuracy of performing tasks and the communication capability of SIoT objects. We study two different TOSS problems based on two different communication requirements, namely *BC-TOSS* and *RG-TOSS*. For BC-TOSS, we propose a polynomial-time algorithm with performance guarantee, and for RG-TOSS, we propose an efficient and effective algorithm that can obtain good solutions in polynomial time. We propose various ordering and pruning strategies for each algorithm to significantly reduce the computation time. Experimental results on real datasets show that our proposed algorithms outperform the other baselines.

8. ACKNOWLEDGEMENTS

This work was supported in part by Ministry of Science and Technology (MOST), Taiwan, under MOST 105-2218-E-007 -032 -MY2, and MOST 105-2218-E-002 -035 -.

9. REFERENCES

[1] B. I. Aydin, Y. S. Yilmaz, Y. Li, Q. Li, J. Gao, and M. Demirbas. Crowdsourcing for multiple-choice question answering. In *IAAI*, 2015.

[2] I.-R. Chen, F. Bao, and J. Guo. Trust-based service management for social internet of things systems. In *TDSC*, 2015.

[3] A. Fast, D. Jensen, and B. N. Levine. Creating social networks to improve peer-to-peer networking. In *KDD*, 2005.

[4] U. Feige, D. Peleg, and G. Kortsarz. The dense k-subgraph problem. In *Algorithmica*, 2001.

[5] A. V. Goldberg. Finding a maximum density subgraph. *Tech. Report No. UCB CSD 84/171*, 1984.

[6] N. Hamadeh, B. Daya, A. Hilal, and P. Chauvet. An analytical review on the most widely used meteorological models in forest fire prediction. In *TAEECE*, 2015.

[7] M. Kargar and A. An. Discovering top-k teams of experts with/without a leader in social networks. In *CIKM*, 2011.

[8] E. A. Kosmatos, N. D. Tselikas, and A. C. Boucouvalas. Integrating rfids and smart objects into a unified internet of things architecture. In *Advances in Internet of Things*, 2011.

[9] T. Lappas, K. Liu, and E. Terzi. Finding a team of experts in social networks. In *KDD*, 2009.

[10] W. Liu, W. Sun, C. Chen, Y. Huang, Y. Jing, and K. Chen. Circle of friend query in geo-social networks. In *DSAA*. Springer, 2012.

[11] R. J. Mokken. Cliques, clubs and clans. *Quality & Quantity*, 13(2):161–173, 1979.

[12] M. Nitti, R. Girau, and L. Atzori. Trustworthiness management in the social internet of things. *TKDE*, 26(5):1253–1266, 2014.

[13] S. B. Seidman. Network structure and minimum degree. In *Social Networks*, 1983.

[14] S. B. Seidman and B. L. Foster. A graph-theoretic generalization of the clique concept*. *Journal of Mathematical Sociology*, 6(1):139–154, 1978.

[15] C.-Y. Shen, D.-N. Yang, W.-C. Lee, and M.-S. Chen. Trustworthiness management in the social internet of things. *TKDD*, 10(47), 2016.

[16] H.-H. Shuai, D.-N. Yang, P. S. Yu, and M.-S. Chen. Willingness optimization for social group activity. In *VLDB*, 2014.

[17] M. Sozio and A. Gionis. The community-search problem and how to plan a successful cocktail party. In *KDD*, 2010.

[18] J. Surowiecki. *The wisdom of crowds*. Doubleday, 2004.

[19] S. Wasserman and K. Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[20] D.-N. Yang, C.-Y. Shen, W.-C. Lee, and M.-S. Chen. On socio-spatial group query for location-based social networks. In *KDD*, 2012.

[21] L. Yao, Q. Z. Sheng, A. H. Ngu, H. Ashman, and X. Li. Exploring recommendations in internet of things. In *SIGIR*, 2014.

[22] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman. Sybilguard: defending against sybil attacks via social networks. In *SIGCOMM*, 2006.

[23] Q. Zhu, H. Hu, J. Xu, and W.-C. Lee. Geo-social group queries with minimum acquaintance constraint. *arXiv preprint arXiv:1406.7367*, 2014.