On Socio-Spatial Group Query for Location-Based Social Networks

De-Nian Yang
Academia Sinica
Taipei, Taiwan
dnyang@iis.sinica.edu.tw

Chih-Ya Shen
National Taiwan Univ.
Taipei, Taiwan
chihya@arbor.ee.ntu.edu.tw

Wang-Chien Lee
The Penn State Univ.
State College, PA, USA
wlee@cse.psu.edu

Ming-Syan Chen
National Taiwan Univ.
Taipei, Taiwan
mschen@cc.ee.ntu.edu.tw

ABSTRACT

Challenges faced in organizing impromptu activities are the requirements of making timely invitations in accordance with the locations of candidate attendees and the social relationship among them. It is desirable to find a group of attendees close to a rally point and ensure that the selected attendees have a good social relationship to create a good atmosphere in the activity. Therefore, this paper proposes Socio-Spatial Group Query (SSGQ) to select a group of nearby attendees with tight social relation. Efficient processing of SSGQ is very challenging due to the tradeoff in the spatial and social domains. We show that the problem is NP-hard via a proof and design an efficient algorithm SSGSelect, which includes effective pruning techniques to reduce the running time for finding the optimal solution. We also propose a new index structure, Social R-Tree to further improve the efficiency. User study and experimental results demonstrate that SSGSelect significantly outperforms manual coordination in both solution quality and efficiency.

Categories and Subject Descriptors

H.2.4 [**Systems**]: Query processing

General Terms

Algorithms

Keywords

Query Processing, Spatial Indexing, Social Networks

1. INTRODUCTION

With the emergence of GPS-enabled mobile devices and Web 2.0 technology, mobile users can easily capture and upload their own locations to various location-based social networking (LBSN) applications, such as Loopt, Geomium, Buddy Beacon, FindMe, Facebook Place, BrightKite, Meetup, and Google Map. All these LBSN applications allow users to share their location information with friends and thus, as we envisage, may potentially enable instant organization of impromptu activities, e.g., having a dinner with friends nearby. However, with today's technology, the selection of invitees for these events still requires significant *manual* coordination. In this work, we aim to develop new techniques for an impromptu activity planning service to alleviate this effort.

Due to the inherited nature as an LBSN application, such an impromptu activity planning service needs to consider the following *spatial* and *social* factors: i) the selected friends need to be close enough to a rally point in order to minimize the waiting time for an activity; and ii) the selected friends need to have a good social relationship to create a good atmosphere in the activity. In other words, challenges faced in organizing impromptu activities lie in meeting the requirements of making *timely* invitations in accordance with the *locations* of candidate invitees and the *social* relationship among them. Notice that friends located nearby may not have tight social relationships with each other, while close friends may not happen to located near an intended rally point. Moreover, when the number of intended invitees increases, the invitation process becomes more complicated and thereby too tedious to coordinate manually. Therefore, efficient search algorithms that automatically suggest the most suitable attendees for an activity is a mandate for the impromptu activity planning service.

As the first step towards the aforementioned impromptu activity planning service, we propose a new query, namely *Socio-Spatial Group Query (SSGQ)*, aiming to find a set of most suitable candidates for a planned activity by taking into account certain spatial and social constraints. In this work, we assume that the service provider has access to the social connectivity of its users and their locations.[1] Thus, given a social graph consisting of socially connected users and their (published) locations, an SSGQ, specified a rally point q, the number of activity invitees p, and the average number of unfamiliar people an invitee may have k, returns a set of p invitees from the social graph of the activity initiator such that the average spatial distance for each invitee to the rally point is minimized. It is worth noting that the query incorporates a social constraint (denoted by k), to govern the unfamiliarity between invitees. Each attendee on average can have no social relationship with at most k other invitees. With a proper setting of k, the tightness of social relationships among invitees (and hopefully activity atmosphere) are ensured. As such, the SSGQ can facilitate

KDD'12, August 12–16, 2012, Beijing, China.
Copyright 2012 ACM 978-1-4503-1462-6 /12/08 ...$15.00.

[1] The privacy issues on the locations and social connectivity of users are very important but out of scope of this study. A simple way is to allow each user to specify a friend list and time intervals, such that only the query issued from the friends in the list during those intervals can access the location information and social connectivity of the user.

users to organize different kinds of activities based on both the specified spatial and social conditions.

Note that SSGQ can be generalized for supporting other spatial/social relevant planning services that do not require a specific initiator. For example, when an user queries a nearby restaurant using a mobile device, an SSGQ may be issued to automatically recommend a group of socially close friends near the restaurant. Moreover, location-based advertisements can leverage SSGQ to a group of friends for a preferred restaurant in order to push mobile coupon, which is beneficial to both customers and restaurants.[2] Finally, the SSGQ is particularly useful for coordinating search and rescue operations in some scenarios of emergency and disasters, such as earthquakes and tsunamis. As social relationship, e.g., previous co-working experiences of the searchers, and spatial factor, e.g., distance for searchers to reach the disaster area, are important for success of rescue operations, the SSGQ can be employed to obtain the most suitable searchers.

The problem of processing SSGQ efficiently is challenging due to the tradeoff in the spatial distance and the social relationships. A simple strategy for processing SSGQ is to enumerate all possible groups of p invitees, eliminate those that do not satisfy the social constraints, and then return the group with minimal total spatial distance. This approach needs to evaluate C_p^f candidate groups, where f is the total number of candidate attendees, in the solution space and thus is not desirable. Note that, as we will show later, the problem of processing SSGQ is NP-hard. Nevertheless, due to the importance of spatial and social factors considered in SSGQ, we exploit the spatial and social constraints specified in SSGQ while forming the candidate groups to alleviate the processing cost. Our idea is to expand the solution space in a systematical way such that we can efficiently find the solution without examining all candidate groups in details. In other words, we incrementally select the invitees by giving priority to those (a) who are located close to the rally point and (b) who are close friends. Finding a group of friends who best meet both conditions of (a) and (b) is not trivial. An algorithm that prioritizes on (a) is inclined to generate candidate groups with small total spatial distance quickly but it does not always come up with a solution that satisfies the familiarity constraint, especially those activities specified with a small k. On the other hand, an algorithm that prioritizes on (b) may quickly find a group of friends who know each other very well but do not easily satisfy the minimum total spatial distance. In other words, the challenge in finding optimal SSGQ answer comes from the dilemma of (1) reducing the total spatial distance[3] and (2) ensuring that the solution satisfying the familiarity constraint.

Specifically, in this paper, we first prove that SSGQ is an NP-hard problem but, fortunately, while the problem is very challenging, it is still tractable since the size of p is relatively small in most impromptu activities. Therefore, to efficiently process this query, we design a new algorithm, called *SSGSelect*, to find the optimal solution. We design various strategies, Distance Ordering, Socio-Spatial Ordering, Distance Pruning, and Familiarity Pruning, to effectively reduce the processing time. During the selection of each attendee, we address both the spatial distance and social connectivity of the attendee, together with the characteristics of the social network for other candidate attendees that have not been considered, in order to prune unnecessary search space and find the optimal solution efficiently.

The above strategies incrementally consider and examine each candidate attendee. A potential approach to further improve the efficiency is to select multiple candidate attendees jointly, because the optimal solution can be constructed with fewer iterations. When an attendee close to the rally point is chosen, nearby candidate attendees are potential to be good candidates for joint selection due to their smaller spatial distances to the rally point. Nearby candidate attendees can be efficiently identified by leveraging the spatial index structure, such as R-Tree, since they are usually located in the same Minimum Bounding Rectangle (MBR) of R-Tree. Nevertheless, choosing *all* candidate attendees in the same MBR may not be able to generate a solution following the familiarity constraint with k since the social connectivity among them is not carefully examined. To effectively tackle this issue, we propose Social R-Tree (SR-Tree), which incorporates and summarizes the social information for the candidate attendees in an MBR. In SR-Tree, all candidate attendees in each MBR are indexed into multiple social clusters with different social densities to support SSGQ with different k.

With the proposed SR-Tree, SSGSelect is enhanced by a new strategy, Joint Insertion, which chooses multiple candidate attendees simultaneously to reduce the number of required iterations for finding the optimal solution. We implement SSGSelect in Facebook and conduct user study on 206 people. The contributions of this paper are summarized as follows.

- We formulate a useful query for social networking application, namely, SSGQ, to obtain the optimal set of attendees. SSGQ can be used to plan for various types of activities by specifying the familiarity constraint k. We prove that the problem is NP-hard.
- To efficiently process SSGQ, we propose SSGSelect with various strategies, including Distance Ordering, Socio-Spatial Ordering, Distance Pruning, and Familiarity Pruning, for finding the optimal solution. In addition, we design a new index structure, named SR-Tree, together with Joint Insertion, to simultaneously select multiple nodes at each iteration for finding the optimal solution in smaller time.
- We implement SSGSelect in Facebook and conduct an user study with 206 people. The results demonstrate that SSGSelect significantly outperforms manual coordination in both solution quality and efficiency.

The rest of this paper is summarized as follows. Section 2 introduces the related works. Section 3 formulates SSGQ and prove that it is NP-hard. We design algorithm SSGSelect in Section 4 and enhance it with the proposed SR-Tree in Section 5. Results of the user study and experiments are shown in Section 6, and we conclude this paper in Section 7.

2. RELATED WORK

Some LBSN applications, e.g., Meetup, have been developed for activity coordination. However, an activity initiator still needs to specify candidate attendees of the activity. This is undesirable because social familiarity is not

[2] Notice that location-based advertisement and social network-based advertisements have been exploited by websites, such as Google Map and Livingsocial.com. However, the two aspects have not been considered in an integrated fashion.

[3] Reducing the maximum spatial distance of an attendee is an alternative. This paper proposes to minimize the total spatial distance because many social activities can begin or warm up when most attendees arrives, and a few distant attendees are allowed to arrive late.

ensured, while the waiting time is not minimized, either. Manual assignment of candidate attendees is tedious and time-consuming for the initiator, especially for a large activity. Thus, SSGQ is very useful for such sites which recommends suitable attendees with efficient query processing.

In recent works, several researches on finding groups of socially connected members, e.g., team formation [2][3], community search [4], and Social-Temporal Group Query [7], have been reported in the literature. Nevertheless, their research context and objectives are totally different from our study that explores both the spatial and social dimensions in finding a group for impromptu activity planning. Specifically, team formation [2][3] finds a group of experts with the required skills, while aiming to minimize the communications cost between these experts. Community search [4] finds a compact community that contains particular members, aiming to minimize the total degree in the community. Social-Temporal Group Query [7] checks the available times of attendees to find the group with the most suitable activity time. To the best knowledge of the authors, researches on finding groups that consider constraints in both the spatial and social dimensions have not appeared. In contrast, our work examines the inter-play in both social and spatial dimensions, with an objective to find a group of mutually familiar attendees such that the total spatial distance to a rally point is minimized. As mentioned, we envisage that our research result can be employed in various LBSN applications for group recommendation.

Relevant to our work, spatial queries for selecting a set of spatial points, aiming to minimize the total spatial distance, have been proposed for various scenarios [5, 6, 8, 9]. However, in these works, the (social) connectivity among the spatial points is not considered. Specifically, given two sets of points P and Q, together with the number of points to be selected k, Group Nearest Neighbor Query [5] finds a set of k points in P such that the total spatial distance of the points to all points in Q is minimized. On the other hand, for a line segment and a set of points, Continuous Nearest Neighbor Search [6] returns the nearest neighbor of each point on the line segment, while the Continuous Visible Nearest Neighbor Queries [8] extends Continuous Nearest Neighbor Search [6] by incorporating the obstacles in the problem design, which may affect the visibility or distance between two points and lead to different results. Meanwhile, Continuous Obstructed Nearest Neighbor Query [9] retrieves the nearest neighbor with regard to the obstructed distance, i.e., the shortest path without crossing any obstacle. Therefore, the above-mentioned queries focus only on the spatial dimension and thereby are not applicable to our scenario of LBSN applications.

3. PROBLEM FORMULATION

Given a social graph $G = (V, E)$, where each vertex $v \in V$ is a candidate attendee with a location l_v, and any two mutually acquainted vertices u and v are connected by an edge $e_{u,v}$. A Socio-Spatial Group Query $SSGQ(p, q, k)$ finds a set F of p vertices from G where the total spatial distance from the vertices in F to the rally point q, i.e., $\sum_{v \in F} d_{v,q}$, is minimized and each vertex on average can share no edge with at most k other vertices in F.

Notice that the three parameters in an SSGQ, i.e., p, q, and k, determine the size of the answer group, spatial constraint, and social constraint of the query, respectively, and thus have significant implications regarding the processing strategy. First, as the size of group p increases, the solution space (which consists of all candidate groups) rapidly grows. Indeed, we prove that processing SSGQ is an NP-hard problem but, fortunately, while the problem is very challenging, it is still tractable since the size of p is relatively small in most cases. Second, the position of q hints that candidate invitees located close to q should be considered with priority as the search criteria aims to minimize the total spatial distance from members of the group to q. Finally, k dictates the tightness of social relationship among members of the group. A small k in SSGQ results in a group of mutually familiar invitees, while a larger k allows more unfamiliarity within the returned group of invitees. The spatial and social constraints can be employed for early pruning of unqualified candidate groups.

In this paper, we aim to develop efficient query processing algorithms for SSGQ with a new index structure, Social R-Tree. Moreover, we also devise several access ordering and pruning strategies to effectively reduce the computation time. In the next section, we design an efficient query processing algorithm by pruning unqualified candidate groups.

In the following, we first prove the hardness result.

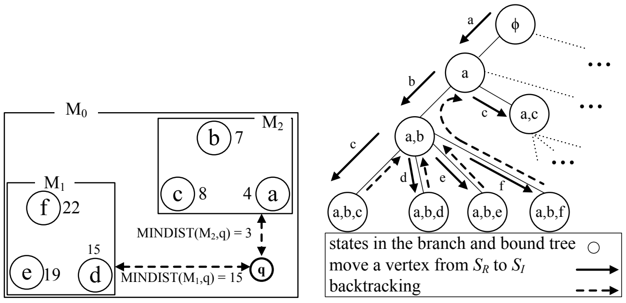
THEOREM 1. *SSGQ is NP-hard.*

PROOF. We prove that SSGQ is NP-hard with the reduction from p-clique. Decision problem p-clique is given a graph G_c to find whether the graph contains a clique, i.e., a complete graph with an edge connecting every two vertices, with p vertices. In SSGQ, we let $G = G_c$, $k = 0$, and $d_{v,q} = 1$ for every vertex $v \in V$. We first prove the necessary condition. If G_c contains a p-clique, there must exist a group with the same vertices in the p-clique such that every person has social relations with all the other attendees of the group, and the total spatial distance is p. We then prove the sufficient condition. If G in SSGQ contains a group with the size as p and k as 0, G_c in problem p-clique must contain a solution with size p, too. The theorem follows. □

4. ALGORITHM DESIGN FOR SSGQ

In this section, we propose Algorithm SSGSelect to find the optimal solution for SSGQ. To guide an efficient exploration of the search space, we design effective strategies to selection new attendees and prune unnecessary search space. Later in Section 5, equipped with SR-Tree, SSGSelect is enhanced by Joint Insert to select multiple attendees at each iteration. Specifically, let S_I and S_R denote the intermediate solution set and the remaining set of candidate attendees, respectively. Initially, S_I is empty, and S_R is V. At each iteration afterward, we select a vertex from S_R to S_I. When S_I includes p vertices and follows the familiarity constraint, S_I is regarded as a feasible solution. Afterward, to improve the feasible solution, our algorithm backtracks to the previous iteration and previous S_I for choosing another vertex from S_R to S_I, while a branch-and-bound tree is maintained to keep track of the exploration process for backtracking.

The selection of the vertex from S_R into S_I at each iteration is critical. It is essential to avoid choosing a vertex that will lead to a large increment of the spatial distance. With this important factor in mind, Distance Ordering selects a vertex with a small spatial distance to the rally point q. To further consider the social connectivity, Socio-Spatial Ordering extends Distance Ordering by jointly considering the spatial and social domains. On the other hand, to trim unnecessary search space, Distance Pruning effectively avoids unnecessary exploration of S_R that will not lead to a better solution, by examining the sum of the current total spatial distance from S_I to q and the lower bound of the distance increment in selecting any vertex from S_R. In addition to



(a) (b)

Figure 1: Example of Distance Ordering and Branch-and-Bound Search.

the above spatial domain, Familiarity Pruning examines social connectiviy of the vertices in S_R and stops choosing any vertex from S_R if eventually it is not able to generate any feasible solution following the familiarity constraint.

Notice that SSGSelect is able to find the optimal solution because Distance Pruning only removes unqualified solutions (the solutions with larger total spatial distances to q), and Familiarity Pruning only trims infeasible solutions (the solutions that do not follow the familiarity constraints). The optimal solution does not belong the above two categories. Moreover, with Socio-Spatial Ordering, SSGSelect exploits both the spatial and social domain information to guide an efficient search for the optimal solution.

4.1 Distance Ordering

At each iteration, Distance Ordering selects the vertex with the minimum spatial distance to q, to minimize the increment of the total spatial distance for S_I. Naturally, a straightforward approach is to sort all vertices in G before the algorithm starts. However, this approach is not scalable for a large social network and is very computation intensive because q is allowed be different for every SSGQ. In contrast, a more efficient approach is leveraging a spatial index structure, such as R-Tree, and its distance browsing strategy [10] with a priority queue for vertex selection. Each element in the priority queue is either a vertex or a Minimum Bounding Rectangle (MBR). The distance of a vertex is its spatial distance to q, while the distance of an MBR M is the $MINDIST(M, q)$, which denotes the minimum distance between q and the border of M [1]. Finding the vertex with the minimum distance can be achieved by iteratively extracting the first element of the priority queue. Initially, the priority queue contains only the root MBR of R-Tree. When the extracted element is an MBR, we traverse the R-Tree and insert all its child MBRs into the priority queue. Therefore, the first extracted vertex from the queue is the one with the minimum distance to q.

For example, Figure 1(a) shows six vertices in an R-Tree, where q denotes the rally point. Assume that $MINDIST$ to q for MBRs M_0, M_1 and M_2 are 0, 15 and 3, respectively, and the actual distances of the vertices to q are marked besides each vertex. Initially, the priority queue contains only $\{M_0\}$. Afterward, we extract M_0 and insert M_1 and M_2 into priority queue. Now the first element of the priority queue is M_2 because $MINDIST(M_2, q) < MINDIST(M_1, q)$. Then, M_2 is popped from the priority queue with its children vertices a, b and c inserted. Now the elements in the priority queue are $\{a, b, c, M_1\}$. Therefore, Distance Ordering first selects and adds a to S_I because its spatial distance to q is the smallest one.

Figure 1(b) illustrates a part of the branch-and-bound tree with $p = 3$, where Distance Ordering iteratively moves the vertex with the minimum distance to q from in S_R to S_I. Notice that S_R is $\{c, d, e, f\}$ during the first visit of the state with $S_I = \{a, b\}$, and c is then selected and added to S_I to create a leaf node with $S_I = \{a, b, c\}$ with $p = 3$ in Figure 1(b). Since the number of vertices in S_I is sufficient for p, Algorithm SSGSelect backtracks from $S_I = \{a, b, c\}$ to $S_I = \{a, b\}$, and S_R now becomes $\{d, e, f\}$ since vertex c was just considered. Similarly, when backtracked from $S_I = \{a, b, d\}$ to $S_I = \{a, b\}$, S_R becomes $\{e, f\}$ since both c and d has been explored.

4.2 Socio-Spatial Ordering

With R-Tree, we demonstrated that Distance Ordering can efficiently select the vertex for S_I to minimize the increment of the total spatial distance to q, i.e., the objective value, at each iteration. Nevertheless, it is also desirable that the selected vertex has good social connectivity to other vertices in S_I in order to follow the familiarity constraint. To incorporate this key factor, Socio-Spatial Ordering enhances Distance Ordering by considering both the spatial distance and social connectivity during the selection of a vertex from S_R to S_I. Intuitively, when a vertex is extracted from the priority queue, we select the vertex and add it to S_I only when the vertex satisfies *Intra-Familiarity Condition.* This condition ensures that S_I together with the selected vertex leads to a mutually familiar group of attendees specified by k. If the vertex does not follow the condition, we extract and examine the next vertex from the priority queue. Therefore, both spatial and social factors are captured in Socio-Spatial Ordering.

To ensure that each selected vertex v has good social connectivity to the vertices in S_I, a simple approach is to restrict that v can be selected only when the number of edges between v and the vertices in S_I exceeds a given threshold. With a larger threshold, a candidate attendee that is familiar with more attendees in S_I is inclined to be chosen, such that it is easier for any solution growing from S_I to satisfy the familiarity constraint. Nevertheless, the connectivity of S_I is not examined during the selection of a vertex in the above approach. Intuitively, if S_I is already a mutually familiar group, it is expected that choosing a vertex v with fewer edges connecting S_I is more proper if it can minimize the total spatial distance. In addition, whether the connectivity of S_I is sufficient depends on k, which can be different in every SSGQ. Therefore, it is desirable to incorporate the connectivity of the vertices in S_I and parameter k during the selection of a vertex from S_R to S_I at each iteration. With the above observations in mind, we propose the following condition for Socio-Spatial Ordering.

Intra-Familiarity Condition (IFC). When a vertex v is extracted from the priority queue, we select the vertex and add it from S_R to S_I only when the vertex satisfies Intra-Familiarity Condition. To effectively consider the social domain, this condition first assumes that v is added to S_I and then examines whether the social connectivity of the new group $S_I \cup \{v\}$ is sufficient according to the criterion k. Specifically, let $F(S_I \cup \{v\})$ denote the *intra-familiarity* of the group,

$$F(S_I \cup \{v\}) = \frac{1}{|S_I \cup \{v\}|} \sum\nolimits_{v \in S_I \cup \{v\}} |N_v|,$$

where N_v is the set of neighbors of v in $S_I \cup \{v\}$. For every attendee in $S_I \cup \{v\}$, intra-familiarity represents the average number of acquainted attendees in $S_I \cup \{v\}$. Vertex v is allowed to be selected and added to S_I if it satisfies IFC specified as follows,

$$F(S_I \cup \{v\}) \geq |S_I \cup \{v\}| - \frac{\widehat{k}\,(|S_I \cup \{v\}|)}{p-1} - 1,$$

where $\widehat{k}$ here is a filtering parameter and set as k initially. Intuitively, when $k = p-1$, the activity allows all attendees to be mutually unfamiliar, and Distance Order in this case is the best strategy. In fact, IFC in this situation becomes $F(S_I \cup \{v\}) \geq -1$, and Socio-Spatial Ordering here is identical to Distance Order. In another extreme case with $k = 0$, IFC becomes $F(S_I \cup \{v\}) \geq |S_I \cup \{v\}| - 1$, and each attendee in $S_I \cup \{v\}$ needs to be acquainted with all the others in $S_I \cup \{v\}$.

It is worth noting that IFC incorporates a filtering parameter $\widehat{k}$, instead of including k directly, in order to properly handle other cases with $0 < k < p-1$. When $k = 0$, if no vertex from S_R can satisfy IFC, it is not necessary to add any vertex v from S_R to S_I because every solution growing from $S_I \cup \{v\}$ cannot follow the familiarity constraint. When $k > 0$, it is worth noting that if no vertex from S_R can satisfy IFC, it does not imply that every solution growing from $S_I \cup \{v\}$ does not have sufficient social connectivity. In contrast, it is possible to find a vertex v in S_R and a solution growing from $S_I \cup \{v\}$ following the familiarity constraint, when other vertices added later bring a sufficient number of edges to the solution. Therefore, for any SSGQ with $k > 0$, Socio-Spatial Ordering sets $\widehat{k}$ as k initially and increases $\widehat{k}$ if no vertex from S_R can satisfy IFC, until at least one vertex follows IFC and is able to be selected for S_I. In other words, IFC first maintains a high criterion for social connectivity by setting $\widehat{k}$ as k, in order to prioritize a vertex leading to sufficient social connectivity. If no vertex from S_R can satisfy such a high criterion, IFC increases $\widehat{k}$ to avoid filtering out any feasible solution, and any vertex in S_R that did not satisfy IFC previously will be examined later with a large $\widehat{k}$.

At each iteration, the vertex chosen by Distance Ordering and Socio-Spatial Ordering may be different. The following example demonstrates that the result obtained by Distance Ordering enjoys a smaller total social distance but does not follow the familiarity constraint. For the six vertices in Figure 1(a) with the corresponding social graph in 2(a), where $p = 3$ and $k = 0$, the exploration of Socio-Spatial Ordering is shown as the thick line in Figure 2(b). In this example, at the beginning $\widehat{k} = 0$ and $S_I = \phi$, since a is the vertex with the minimum spatial distance to q and $F(\phi \cup \{a\}) = \frac{0}{1} \geq 1 - 1 - \frac{0 \cdot 1}{2}$ satisfies IFC, Socio-Spatial Ordering moves vertex a from S_R to S_I first and let $S_I = \{a\}$. However, $F(S_I \cup \{b\}) = \frac{0}{2} < 2 - 1 - \frac{0 \cdot 2}{2}$, which does not satisfy IFC. Therefore, Socio-Spatial Ordering examines vertex c and finds $F(S_I \cup \{c\}) = \frac{1}{2} \cdot 2 = 1 \geq 2 - 1 - \frac{0 \cdot 2}{2}$, which satisfies IFC. Therefore, vertex c is moved into S_I and now we have $S_I = \{a, c\}$. We then try to expand S_I by checking IFC with vertex d. $F(S_I \cup \{d\}) = \frac{1}{3} \cdot 6 = 2 \geq 3 - 1 - \frac{0 \cdot 3}{2}$. Therefore, d is moved to S_I, and $S_I = \{a, c, d\}$ now is a feasible solution. By contrast, Distance Ordering first selects vertex b as shown in the dashed-line in Figure 2(b) and then sequentially constructs four states $\{a, b, c\}$, $\{a, b, d\}$, $\{a, b, e\}$ and $\{a, b, f\}$ in the branch-and-bound tree. Unfortunately all of them cannot follow the familiarity constraint. Therefore, this example illustrates that it is desirable to jointly consider spatial and social domain in order to find a feasible solution for SSGSelect earlier, because this feasible solution is a key factor for the pruning strategy in the next subsection.

4.3 Distance Pruning

It is expected that Socio-Spatial Ordering can acquire the first feasible solution, which follows the familiarity constraint, prior to Distance Ordering because social connectivity is examined during the selection of vertices. With the first feasible solution and its total spatial distance D as a criterion, SSGSelect is able to prune unnecessary search space, in which the total spatial distance of any solution is no better than D. Moreover, to further improving the pruning capability, SSGSelect updates D when a better feasible solution is obtained afterward. As shown in our experimental results in Section 6, pruning of unnecessary search space can significantly reduce the time to find the optimal solution.

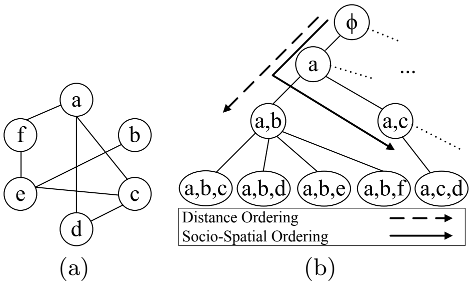


Figure 2: Example of Socio-Spatial Ordering.

For any intermediate solution S_I, there are $p - |S_I|$ vertices required to be selected from S_R to S_I. Apparently, further processing of S_I is unnecessary if choosing the $p - |S_I|$ vertices with the smallest spatial distances to q is not able to generate a solution better than D. Nevertheless, finding the $p - |S_I|$ vertices with the smallest spatial distances for every S_R in the branch-and-bound tree is inclined to incur more delay, especially for a large p. Therefore, Distance Pruning identifies a lower bound and stops processing S_I when the following condition holds,

$$\sum\nolimits_{u \in S_I} d_{u,q} + (p - |S_I|) d_{v_{\min},q} \geq D,$$

where the first term is the total spatial distance from the vertices in S_I to q. For S_R, only the vertex $v_{\min}$ with the smallest spatial distance to q is accessed here, and $(p - |S_I|) d_{v_{\min},q}$ represents a lower bound on the total spatial distance for the above $p - |S_I|$ vertices in S_R. Distance Pruning is computationally efficient since vertex $v_{\min}$ can be extracting from the priority queue in Section 4.1. On the other hand, if the first element of the priority queue is an MBR, Distance Pruning is specified as follows,

$$\sum\nolimits_{v \in S_I} d_{v,q} + (p - |S_I|) MINDIST(M, q) \geq D,$$

where $MINDIST(M, q)$ is the minimum distance between q and the border of M. Thus, the spatial distance of every vertex in M must be no smaller than $MINDIST(M, q)$. That is, $MINDIST(M, q) \leq v_{\min}$, and $(p - |S_I|) MINDIST(M, q)$ is a lower bound on the total spatial distance of the above $p - |S_I|$ vertices in S_R.

Consider an example with $p = 3$ with the spatial locations and social graph shown in Figure 1(a) and 2(a), respectively. After a feasible solution $\{a, c, d\}$ is explored, its total spatial distance 27 is assigned to D. When SSGSelect considers $S_I = \{a, b\}$ and $S_R = \{e, f\}$, since $\sum_{u \in S_I} d_{u,q} + (p - |S_I|) d_{v_{\min},q} = 11 + 1 \cdot 19 = 30 > 27$, Distance Pruning removes states $\{a, b, e\}$ and $\{a, b, f\}$, stops processing $S_I = \{a, b\}$, and backtracks to the previous state accordingly.

4.4 Familiarity Pruning

Distance Pruning trims the search space by stopping processing S_I and backtracks to the previous state in the branch-and-bound tree, when the total spatial distance from each vertex in S_R to q is too large to generate a solution better than the current feasible solution D in hand. In contrast, Familiarity Pruning examines the social connectivity of all possible solutions growing from S_I. Familiarity Pruning stops processing S_I and backtracks to the previous state if every possible solution growing from S_I via the exploration of S_R is not able to satisfy the familiarity constraint.

The challenge of designing a pruning strategy in social domain lies in various social connectivity allowed to appear in all possible solutions. It is expensive to examine each possible solution according to the familiarity constraint. Therefore, to efficiently prune unnecessary search space, Familiarity Pruning derives an upper bound on the social closeness of every possible solution growing from S_R. If the upper bound indicates that each attendee on average is acquainted with fewer than $p-k-1$ other attendees, every possible solution is not able to follow the familiarity constraint.

Specifically, the edges in any solution growing from S_I can be divided into three categories: 1) the set of edges E_I connecting any two vertices in S_I, 2) the edges of edges E_R connecting any two vertices selected from S_R, and 3) the set of edges E_{IR} connecting any two vertices in S_I and the vertices selected from S_R. Apparently, $|E_I|$ is $\frac{1}{2}\sum_{v \in S_I} |N_v^I|$, where N_v^I is the set of acquainted neighbors of v in S_I. Since the selected vertices in S_R is not clear now, a good way is to find an upper bound on $|E_R|$, i.e., $\frac{1}{2}(p-|S_I|)\max_{v \in S_R} |N_v^R|$, where N_v^R is the set of acquainted neighbors of v in S_R. It is an upper bound because the vertex with the maximum degree in S_R is identified, and $(p-|S_I|)$ vertices are selected from S_R. Similarly, an upper bound on $|E_{IR}|$ is $\sum_{v \in S_I} |InterEdge(v)|$, where $InterEdge(v)$ is the set of edges connecting v in S_I to any vertices in S_R.

Notice that the number of edges in a feasible solution is half of the degree sum of all the vertices in the solution, where the degree of a vertex represents the number of acquainted neighbors in the solution. Therefore, with the above three categories of edges, Familiarity Pruning stops processing S_I when the following condition holds,

$$\begin{aligned} &\frac{1}{p}\left[\sum\nolimits_{v \in S_I} |N_v^I| + (p-|S_I|)\max_{v \in S_R} |N_v^R| \right. \\ &\left. +2 \cdot \sum_{v \in S_I} |InterEdge(v)| \right] \\ < \quad &(p-k-1), \end{aligned}$$

where the left-hand-side is an upper bound on the average number of attendees acquainted to each person in any feasible solution growing from S_I. In the above condition, each attendee on average is acquainted with fewer than $p-k-1$ other attendees. Familiarity Pruning stops processing S_I and backtracks to the previous state if every possible solution growing from S_I via the exploration of S_R is not able to satisfy the familiarity constraint.

For the social graph in Figure 2(a) with $p=3$, $k=0$, if $S_I=\{b,d\}$ and $S_R=\{e,f\}$, Familiarity Pruning stops processing S_I because $\frac{1}{3}(0+1\cdot 1+2\cdot 1)=1<(3-0-1)=2$. In other words, moving any vertex from S_R to S_I will never generate a feasible solution following the familiarity constraint.

5. ENHANCED STRATEGY FOR SSGQ

The above strategies efficiently select and examine each candidate attendee. A potential approach to further improve the efficiency is to select multiple candidate attendees jointly, because the optimal solution can be constructed with fewer iterations. When an attendee close to the rally point is chosen, nearby candidate attendees in the same MBR of R-Tree are also potential to be good candidates for joint selection due to their smaller spatial distances to the rally point. Nevertheless, jointly selecting *all* candidate attendees in the same MBR may not be able to generate a solution following the familiarity constraint since the social connectivity among them is not examined. To effectively tackle this issue, we propose Social R-Tree (SR-Tree), in which candidate attendees in each MBR are indexed into multiple social clusters with different social connectivity to support SSGQ with different k. Equipped with SR-Tree, SSGQ reduces the number of iterations required to find the optimal solution with a new strategy, named Joint Insertion, which selects multiple vertices jointly at each iteration for S_I to grow faster.

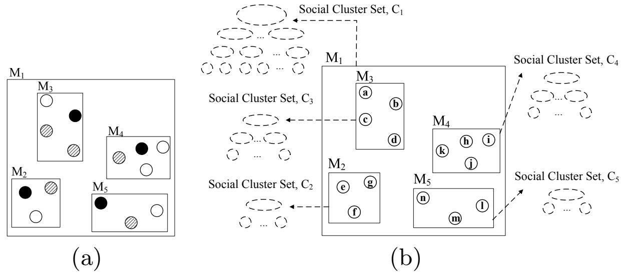


Figure 3: Illustrations of SR-Tree.

5.1 Social R-Tree

In section 4.1, the feature of hierarchical MBRs in R-Tree enables SSGQ to efficiently extract the vertex with the smallest spatial distance to q. It is expected that the vertices in the same MBR are spatially closer to each other. Nevertheless, each vertex may not be familiar with all the others in the same MBR. For example, Figure 3(a) illustrates that only the vertices with the same color are acquainted with each other, and inserting all the vertices in M_1 to S_I is potential to be too aggressive to generate a solution satisfying the familiarity constraint. On the other hand, choosing a group of mutually acquainted vertices is not efficient, especially for a large k, since more vertices in the same MBR can be selected jointly. Thus, it is desirable to identify *social clusters* with different connectivity in order to consider various SSGQ with different k possibly specified by users.

In the following, we propose SR-Tree by finding a set of social clusters C_i for each MBR M_i of R-Tree, as shown in Figure 3(b). C_i for each MBR M_i is constructed in a bottom-up manner. We iteratively combine small social clusters in a lower level to large social cluster in the higher level in Figure 5. At the bottom level (level 0), each cluster is a single vertex. Multiple clusters in level i are combined to generate a new cluster in level $i+1$, while each cluster in level i is allowed to participate in multiple clusters in level $i+1$. For example, cluster $\{a\}$ at level 0 in Figure 5 appears in $\{a,b\}$ and $\{a,d\}$ in level 1. We tailor the social clusters for SSGQ according to the following three requirements.

(1) **Cross MBR Requirement**. To avoid generating duplicated clusters in multiple MBRs, each cluster c must span at least two child MBRs of the current MBR M_i, since a cluster spanning only one child MBR can be generated when we consider the child MBR. Figure 5 shows an example, where clusters $\{a,c\}$, $\{a,e\}$, $\{b,d\}$, $\{b,f\}$, $\{c,e\}$ and $\{d,f\}$ are not generated since the vertices in each of them are located within the same MBR.

(2) **Social Connectivity Requirement.** This require-

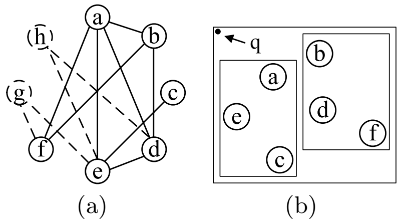


Figure 4: Example of a social network and the corresponding MBRs.

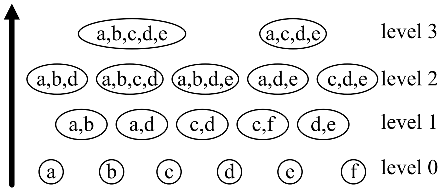


Figure 5: Hierarchically constructing the set of social cluster C_i for MBR M_i.

ment specifies the minimum social connectivity allowed for each generated cluster c. By lowering this requirement, more social clusters can be generated at the cost of more memory consumption. To strike a good balance between efficiency and memory requirement, a good way is referring to parameters k and p appearing in the past SSGQ history. Let P_S denote the average $\frac{k}{p}$ specified in SSGQ of the history. For each vertex v in c, P_S represents the percentage of vertices in c with no edge connecting to v. Therefore, Social Connectivity Requirement specifies that on average each vertex in a social cluster is allowed to share no edge with at most $\lceil |c|\, P_S \rceil$ other vertices in c. Here the ceiling function is adopted such that small social clusters are easier to be generated and then combined to large clusters. Consider an example with the social graph in Figure 4(a) and the corresponding MBRs in Figure 4(b), where vertices g and h are not located in these MBRs. When $P_S = 0.3$, cluster $\{a, b, c\}$ cannot be created in level 1 of Figure 5, because each vertex on average shares no edge with $\frac{4}{3}$ other vertices and exceeds $\lceil |c|\, P_S \rceil = 1$. Similarly, clusters $\{a, c, d\}$, $\{a, c, d, f\}$ and $\{c, d, f\}$ do not have sufficient social connectivity to be created.

(3) **Spatial Distance Requirement.** To avoid the case that the vertices of a social cluster is distributed diversely in a large MBR, it is necessary to limit the spatial distribution of a cluster, and Spatial Distance Requirement ensures that the average distance from a vertex to the *geometric median* of c must not exceed a threshold T_d. Given the coordinates (x_j, y_j) of each vertex v_j in c, the coordinates (x_m, y_m) of the geometric median m is $(x_m, y_m) = \arg\min_{(x'_m, y'_m)} \sum_{v_j \in c} \sqrt{(x'_m - x_j)^2 + (y'_m - y_j)^2}$. The geometric median is regarded as a spatial index in SR-Tree and plays an important role in Join Insertion to prioritize the selection of a social cluster. In Figure 5, cluster $\{e, f\}$ cannot appear in level 1 if T_d is identical to the distance from a to b.

5.2 Joint Insertion

To improve the efficiency of SSGSelect, Joint Insertion selects a social cluster with multiple vertices and moves them from S_R to S_I. A Joint Insertion operation can be regarded as a shortcut in the branch-and-bound tree, such that a good feasible solution D can be acquired earlier for Distance Pruning to remove unnecessary vertices that are not able to generate a better solution. Therefore, equipped with SR-Tree and Joint Insertion, SSGSelect can find the optimal solution with smaller time.

Table 1: $d_{m,q}$ and corresponding actual average spatial distance to q.

	$\{a, b\}$	$\{a, b, d, e\}$	$\{a, b, c, d, e\}$	$\{a, d\}$
$d_{m,q}$	9.6	10	11.2	11.3
$\sum_{v \in c} \frac{d_{v,q}}{\lvert c \rvert}$	9.8	10.8	11.7	11.8

Similar to Socio-Spatial Ordering, since it is desirable to choose a dense social cluster with a small average spatial distance to q, both the social and spatial domains are examined during the selection of a social cluster. Nevertheless, with only the spatial location of each vertex in c, it is expensive for a large cluster to find the average distance to q, because the spatial distance from each vertex to q specified at this query needs to be first calculated and then summed up on-line, while q is allowed to be different in each SSGQ. Therefore, Joint Insertion leverages the geometric median m in Section 5.1, which can be computed off-line and utilized for each query, and regards $d_{m,q}$ as the estimation of the average spatial distance from c to q. Note that Joint Insertion does not consider any cluster with $|S_I| + |c| > p$ or $S_I \cap c \neq \varnothing$. The reason is that the geometric median of c in the case with $S_I \cap c \neq \varnothing$ cannot correctly estimate the average spatial distance for the vertices to be jointly inserted, since some of them is already in S_I.

The social clusters are sorted according to the spatial distance from the geometric median of each cluster to q. Similar to Socio-Spatial Ordering, the first social cluster that satisfies Intra-Familiarity Condition (IFC) in Section 4.2 is selected and inserted to S_I, while IFC here examines $S_I \cup c$, instead of $S_I \cup \{v\}$.

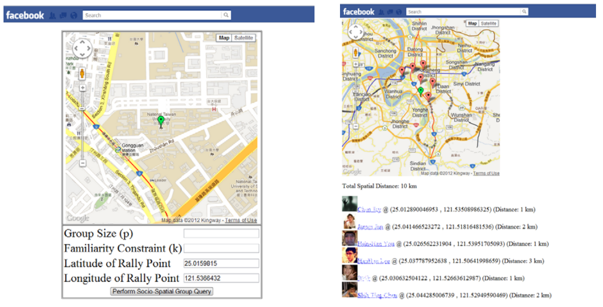
Consider an example with the social graph and the corresponding MBRs in Figure 4, where $S_I = \{g, h\}$, $p = 6$, $k = 3$ and the rally point q is shown in Figure 4(b). Assume the total spatial distance from S_I to q is 13, and Table 1 lists the clusters with $d_{m,q}$, together with their corresponding actual average spatial distance to q. Joint Insertion first finds out that $\{a, b\}$ cannot be moved to S_I because it does not satisfy IFC with $\widehat{k} = 3$, i.e., $F(S_I \cup \{a, b\}) = \frac{1}{4} \cdot 2 = 0.5 < 4 - 1 - \frac{3 \cdot 4}{5}$. We then consider the intra-familiarity of $\{a, b, d, e\}$, $F(S_I \cup \{a, b, d, e\}) = \frac{1}{6} \cdot 16 \geq 6 - 1 - \frac{3 \cdot 6}{5}$, which satisfies IFC. Therefore, Joint Insertion simultaneously moves a, b, d, and e from S_R to S_I at one iteration, which produces a feasible solution with the total spatial distance as $13 + 10.8 \cdot 4 = 56.2$.

Join Insertion of c to S_I represents a shortcut in the branch-and-bound tree to efficiently find a feasible solution for Distance Pruning. After finding the feasible solution, SSGSelect backtracks along the original shortcut back to S_I and explores the search space in the same way described in Section 4 afterward. It is worth noting that some vertices in c needs to be considered again in order to construct another feasible solution with those vertices, S_I, and other vertices not appearing in $S_I \cup c$, to ensure that SSGSelect is able to find the optimal solution eventually.

6. EXPERIMENTAL RESULTS

6.1 Experiment Setup

We implement SSGSelect in Facebook. Figure 6(a) shows the input page of SSGQ to specify p, k, and the coordinate of q, which is captured by clicking a location in the map. The result of SSGSelect is displayed in Figure 6(b) with the names of the selected attendees and their locations. We



Group Size (p)
Familiarity Constraint (k)
Latitude of Rally Point 25.0159815
Longitude of Rally Point 121.5366432
Perform Socio-Spatial Group Query

(a)

Total Spatial Distance: 10 km

Chun Ivy @ (25.012890046953 , 121.53508986325) (Distance: 1 km)
James Jan @ (25.041466523272 , 121.51816481536) (Distance: 2 km)
Hsin-Han You @ (25.026562231904 , 121.53951705093) (Distance: 1 km)
Hanlin Lee @ (25.037787952638 , 121.50641998659) (Distance: 3 km)
[illegible] @ (25.030632504122 , 121.52663612987) (Distance: 1 km)
Shih Ting Chen @ (25.044285006739 , 121.52949590469) (Distance: 2 km)

(b)

Figure 6: Implementation of SSGQ in Facebook.

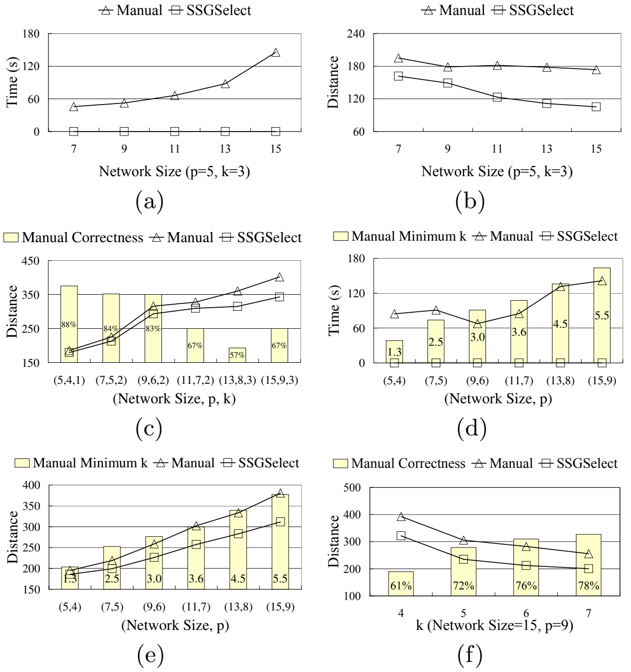
invite 206 people from various communities, e.g., schools, government, and business to join our user study, which compares the solution quality and the time to answer SSGQ by manual coordination and SSGSelect. Each of them answers 24 SSGQ problems with the social graphs extracted from their social networks in Facebook, together with their spatial locations sampling from their Facebook Checkin records shared to the user study. The 24 problems span various p and network sizes. Different k are assigned in the first 12 problems, while k is not specified in the other 12 problems, to let each user freely select p people for finding out the familiarity preferred by each person in activities with different p. In addition to the real dataset *DataSet_FB* from the 206 people, we evaluate the performance and the solution quality of SSGSelect on a large real dataset, *DataSet_Large*, which is obtained by crawling Foursquare [11], one of the most representative LBSNs, for a month. *DataSet_Large* contains both the social and spatial information of 153577 individuals. The rally point q is assigned randomly, and we measure 50 samples in each scenario. Our algorithm is implemented in an IBM 3650 server with Linux Ubuntu 4.2.4, two Quadcore Intel X5450 3.0 GHz CPUs, and 8 GB RAM.

To evaluate the effectiveness of the proposed strategies in SSGSelect, three additional algorithms, Baseline, SSGSelectNoJI, and SSGHeuri(i) are evaluated here. Baseline is a brute-force algorithm that enumerates every possible solution. SSGSelectNoJI is identical to SSGSelect with SR-Tree and Joint Insertion removed. SSGHeuri(i) is a heuristic algorithm used for *DataSet_Large* and outputs the i-th feasible solution obtained by SSGSelect.

6.2 User Study

Figures 7(a)-(f) compare manual coordination and SSGSelect to answer SSGQ in the user study. Figure 7(a) compares the time to find the solutions in different scenarios. The result indicates that SSGQ is challenging for manual coordination, especially for a large network size. In contrast, SSGSelect obtains the optimal solution with less than 1 second. Figure 7(b) with $p = 5$ and $k = 3$ demonstrates that the solutions from manual coordination lead to larger spatial distance and thereby are not optimal. With a larger network size, i.e., more friends nearby, it is easier to find a group of attendees with a smaller total spatial distance to q. In addition, the solution quality in Figure 7(c) shows that even in $p = 5$, each solution obtained by manual coordination is not guaranteed to follow the familiarity constraint, according to the correctness rate shown in Figure 7(c), because it is very challenging for each person to jointly minimize the total spatial distance and ensure the familiarity constraint, and the correctness rate drops dramatically as the network size increases.

In Figures 7(d) and (e), we let each user freely select p people to find out the familiarity preferred by each person in activities with different p. The minimum k here represents the smallest k for each solution to follow the familiarity constraint. With this parameter extracted from the manual solution, we regard it as an input parameter for a SSGQ in the same social network, and the results demonstrate that SSGSelect can find a better solution following the same k with smaller time. In other words, even when an user does not specify k, it is possible to analyze the previous manual coordination results and find out a suitable k for the user, such that SSGSelect is able to find the optimal solutions in each query afterward.



(a) (b)

(c) (d)

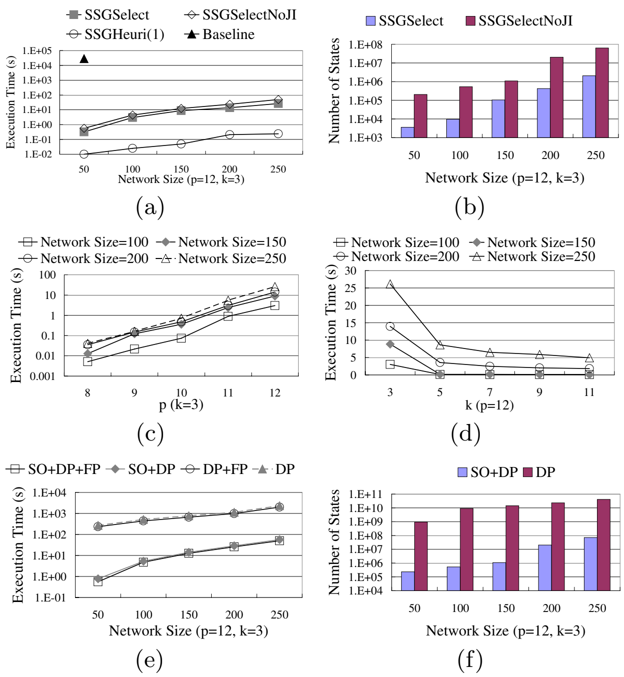
(e) (f)

Figure 7: Results of user study.

Figure 7(f) with network size as 15 and p as 9 compares the results of different k. As k decreases, the correctness rate of manual coordination drops because it becomes more difficult for an user to find a tighter social group with the same number of attendees. Moreover, the solution obtained by manual coordination is still worse than the solution of SSGSelect even with a loose requirement on social connectivity, i.e., a large k.

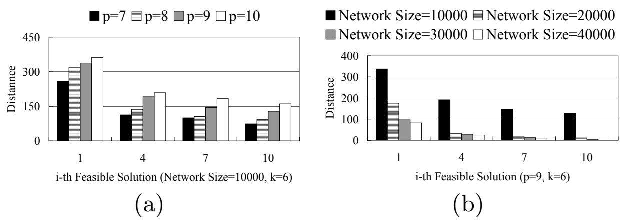
6.3 Performance Evaluation

Figures 8(a)-(b) evaluate the efficiency of different algorithms on *DataSet_FB*. Figure 8(a) compares the execution time of the four algorithms with different network sizes. Since the search space of Baseline is enormous, Baseline is unable to return a solution within 12 hours when the network size is larger than 50. Therefore, there is only a single node for Baseline in Figure 8(a). On the other hand, with Socio-Spatial Ordering, Distance Pruning, and Familiarity Pruning, SSGSelect and SSGSelectNoJI both return the optimal solution in much smaller time. Furthermore, SSGSelect enhanced by SR-Tree and Joint Insertion outperforms SSGSelectNoJI because it can find feasible solutions in a shorter time for Distance Pruning to remove unnecessary search space earlier. Figure 8(b) compares the number of explored states in the branching-and-bound tree required to find a feasible solution of SSGSelect and SSGSelectNoJI.



(a) (b) (c) (d) (e) (f)

Figure 8: Experimental results on *DataSet_FB*.



(a) (b)

Figure 9: Experimental results on *DataSet_Large*.

It demonstrates that SSGSelect only needs 2% of the states in SSGSelectNoJI to find a feasible solution.

In addition to the network size, we compare SSGSelect with different p and k in Figures 8(c) and (d). Figure 8(c) indicates that the execution time increases as p grows, because SSGSelect in this case needs to explore a larger search space to find the optimal solution. On the other hand, Figure 8(d) shows that a larger k leads to a smaller execution time because it becomes easier to obtain feasible solutions for Distance Pruning to trim the search space. Figures 8(e) and (f) analyze the proposed strategies in Section 4. The result indicates that Socio-Spatial Ordering (SO) plays a crucial role to reduce the execution time for at least 40 times. This is because Socio-Spatial Ordering considers both spatial and social domains and thereby is able to guide the efficient search of the feasible solutions and optimal solution with fewer states explored in the branch-and-bound tree, as shown in Figure 8(f).

Figure 9 shows the solution quality of SSGHeuri(i) on the real dataset, *DataSet_Large*. Figure 9(a) demonstrates that the solution converges quickly as i increases, and SSGHeuri(i) can obtain good solutions within the first few feasible solutions. In Figure 9(b), we compare the i-th feasible solutions with different network sizes. The distances decrease when the network size grows since it is expected that a larger network includes more candidate nodes with smaller spatial distances.

7. CONCLUSION AND FUTURE WORK

To the best of our knowledge, there is no real system and existing work in the literature that addresses the issues of automatic activity planning based on social and spatial relationship of activity attendees. In this paper, we define a useful query, namely, SSGQ, to obtain the optimal set of attendees. We show that the problem is NP-hard and devise an efficient algorithm, namely, SSGSelect to process SSGQ. Various strategies, including Distance Ordering, Socio-Spatial Ordering, Distance Pruning, and Familiarity Pruning are explored to prune unnecessary search space and obtain the optimal solution efficiently. Moreover, with SR-Tree and Joint Insert, SSGSelect is improved and requires a smaller time to find the optimal solution. The algorithm is implemented in Facebook, and our user study demonstrates that the proposed algorithm significantly outperforms manual coordination in both solution quality and efficiency.

User study also brings useful suggestions to further enrich SSGQ as our future work. For example, since each candidate attendee is associated with multiple attributes in Facebook, such as age, gender, interest, company name, etc, those attributes can be specified as the input parameters for SSGQ to first filter unsuitable candidate attendees. A more interesting extension is to allow each user to specify the minimum number of attendees with each attribute value required to be selected, such as ensuring an activity with the same number of male and female attendees when considering the gender attribute. Some users also suggest us to integrate the proposed query processing system with automatically delivery of invitation messages and navigation instructions.

8. REFERENCES

[1] N. Roussopoulos, S. Kelley and F. Vincent. Nearest Neighbor Queries. *SIGMOD*, 1995.

[2] T. Lappas, K. Liu, and E. Terzi. Finding a Team of Experts in Social Networks. *KDD*, 2009.

[3] C.-T. Li and M.-K. Shan. Team Formation for Generalized Tasks in Expertise Social Networks. *SocialCom*, 2010.

[4] M. Sozio and A. Gionis. The Community-Search Problem and How to Plan a Successful Cocktail Party. *KDD*, 2010.

[5] D. Papadias, Q. Shen, Y. Tao and K. Mouratidis. Group Nearest Neighbor Queries. *ICDE*, 2004.

[6] Y. Tao, D. Papadias and Q. Shen. Continuous Nearest Neighbor Search. *VLDB*, 2002.

[7] D.-N. Yang, Y.-L. Chen, W.-C. Lee and M.-S. Chen. On Social-Temporal Group Query with Acquaintance Constraint. *VLDB*, 2011.

[8] Y. Gao, B. Zheng, W.-C. Lee, G. Chen. Continuous Visible Nearest Neighbor Queries. *EDBT*, 2009.

[9] Y. Gao and B. Zheng. Continuous Obstructed Nearest Neighbor Queries in Spatial Databases. *SIGMOD*, 2009.

[10] G. Hjaltason and H. Samet. Distance Browsing in Spatial Databases. *TODS*, 1999.

[11] M. Ye, P. Yin, W.-C. Lee and D.-L. Lee. Exploiting Geographical Influence for Collaborative Point-of-Interest Recommendation. *SIGIR*, 2011.